NAVAL POSTGRADUATE SCHOOL

MONTEREY, CALIFORNIA

DISSERTATION

TRAFFIC-ADAPTIVE, FLOW-SPECIFIC MEDIUM ACCESS FOR WIRELESS NETWORKS

by

T. Owens Walker III

September 2009

Dissertation Supervisors: Murali Tummala
John McEachen

Approved for public release; distribution is unlimited

REPORT DOCUMENTATION PAGE			*Form Approved OMB No. 0704-0188*
Public reporting burden for this collection of information is estimated to average 1 hour per response, including the time for reviewing instruction, searching existing data sources, gathering and maintaining the data needed, and completing and reviewing the collection of information. Send comments regarding this burden estimate or any other aspect of this collection of information, including suggestions for reducing this burden, to Washington headquarters Services, Directorate for Information Operations and Reports, 1215 Jefferson Davis Highway, Suite 1204, Arlington, VA 22202-4302, and to the Office of Management and Budget, Paperwork Reduction Project (0704-0188) Washington DC 20503.			
1. AGENCY USE ONLY ***(Leave blank)***	**2. REPORT DATE** September 2009	**3. REPORT TYPE AND DATES COVERED** Dissertation	
4. TITLE AND SUBTITLE: Traffic-adaptive, Flow-specific Medium Access for Wireless Networks		**5. FUNDING NUMBERS**	
6. AUTHOR: T. Owens Walker III			
7. PERFORMING ORGANIZATION NAME(S) AND ADDRESS(ES) Naval Postgraduate School Monterey, CA 93943-5000		**8. PERFORMING ORGANIZATION REPORT NUMBER**	
9. SPONSORING / MONITORING AGENCY NAME(S) AND ADDRESS(ES) N/A		**10. SPONSORING / MONITORING AGENCY REPORT NUMBER**	
11. SUPPLEMENTARY NOTES The views expressed in this thesis are those of the author and do not reflect the official policy or position of the Department of Defense or the U.S. Government.			
12a. DISTRIBUTION / AVAILABILITY STATEMENT Approved for public release; distribution is unlimited		**12b. DISTRIBUTION CODE**	

13. ABSTRACT (maximum 200 words)

This dissertation pioneers the concept of traffic-adaptive, flow-specific medium access and demonstrates that it outperforms contention, non-contention and hybrid techniques. The novel traffic-adaptive Cooperative Wireless Sensor Network Medium Access Control (CWS-MAC) scheme is proposed and shown to provide better throughput and delay performance than slotted, non-persistent carrier sense multiple access (CSMA), upon which the IEEE 802.11 standard is based, and time division multiple access (TDMA). A general model for traffic-adaptive, flow-specific medium access control is developed and hybrid, contention and non-contention schemes are shown to be special cases.

This work also compares the energy efficiency of centralized and distributed solutions and proposes an energy-efficient version of traffic-adaptive CWS-MAC that includes an adaptive sleep cycle coordinated through the use of preamble sampling. A preamble sampling probability parameter is introduced to manage the trade-off between energy efficiency and throughput and delay performance.

Finally, this research quantifies the effect of large propagation delays on contention and contention-free medium access and proposes a flow-specific medium access scheme for networked satellite systems that is based on traffic-adaptive CWS-MAC and is shown to outperform both CSMA- and TDMA-based solutions.

14. SUBJECT TERMS Flow-specific; Traffic-adaptive; Cross layer; Medium access control; Wireless; Energy-efficiency; Preamble sampling; Networked satellite systems			**15. NUMBER OF PAGES** 264
			16. PRICE CODE
17. SECURITY CLASSIFICATION OF REPORT Unclassified	**18. SECURITY CLASSIFICATION OF THIS PAGE** Unclassified	**19. SECURITY CLASSIFICATION OF ABSTRACT** Unclassified	**20. LIMITATION OF ABSTRACT** UU

NSN 7540-01-280-5500 Standard Form 298 (Rev. 2-89)
Prescribed by ANSI Std. 239-18

THIS PAGE INTENTIONALLY LEFT BLANK

Approved for public release; distribution is unlimited

TRAFFIC-ADAPTIVE, FLOW-SPECIFIC MEDIUM ACCESS FOR WIRELESS NETWORKS

T. Owens Walker III
Commander, United States Navy
B.S.E.E., Cornell University, 1987
M.S.E.E, Naval Postgraduate School, 1995

Submitted in partial fulfillment of the requirements for the degree of

DOCTOR OF PHILOSOPHY IN ELECTRICAL ENGINEERING

from the

NAVAL POSTGRADUATE SCHOOL
September 2009

Author: _____
T. Owens Walker III

Approved by:

_____	_____
Murali Tummala Professor of Electrical and Computer Engineering Dissertation Committee Chair and Dissertation Co-Advisor	John McEachen Professor of Electrical and Computer Engineering Dissertation Co-Advisor
_____	_____
Herschel Loomis Professor of Electrical and Computer Engineering	Roberto Cristi Professor of Electrical and Computer Engineering
_____	
J. Bret Michael Professor of Computer Science	

Approved by: _____
Jeffrey B. Knorr, Chair, Department of Electrical and Computer Engineering

Approved by: _____
Douglas Moses, Vice Provost for Academic Affairs

THIS PAGE INTENTIONALLY LEFT BLANK

ABSTRACT

This dissertation pioneers the concept of traffic-adaptive, flow-specific medium access and demonstrates that it outperforms contention, non-contention and hybrid techniques. The novel traffic-adaptive Cooperative Wireless Sensor Network Medium Access Control (CWS-MAC) scheme is proposed and shown to provide better throughput and delay performance than slotted, non-persistent carrier sense multiple access (CSMA), upon which the IEEE 802.11 standard is based, and time division multiple access (TDMA). A general model for traffic-adaptive, flow-specific medium access control is developed and hybrid, contention and non-contention schemes are shown to be special cases.

This work also compares the energy efficiency of centralized and distributed solutions and proposes an energy-efficient version of traffic-adaptive CWS-MAC that includes an adaptive sleep cycle coordinated through the use of preamble sampling. A preamble sampling probability parameter is introduced to manage the trade-off between energy efficiency and throughput and delay performance.

Finally, this research quantifies the effect of large propagation delays on contention and contention-free medium access and proposes a flow-specific medium access scheme for networked satellite systems that is based on traffic-adaptive CWS-MAC and is shown to outperform both CSMA- and TDMA-based solutions.

THIS PAGE INTENTIONALLY LEFT BLANK

TABLE OF CONTENTS

THIS PAGE INTENTIONALLY LEFT BLANK

LIST OF FIGURES

LIST OF TABLES

THIS PAGE INTENTIONALLY LEFT BLANK

LIST OF ACRONYMS AND ABBREVIATIONS

A/D.....analog/digital
ACK .....acknowledgement
AEA .....Adaptive Election Algorithm
AI-LMAC .....Adaptive, Information-centric LMAC
AP .....access point
as-s .....asymptotically second-order self-similar
ATIM .....ad hoc traffic indication map
ATM.....Asynchronous Transfer Mode
AWGN .....additive white Gaussian noise
BER.....bit error rate
B-MAC .....Berkeley Media Access Control
BMDS .....Ballistic Missile Defense System
CCA .....clear channel assessment
CDMA.....code division multiple access
CM .....Control Message
CPU.....central processing unit
CSI .....channel state information
CSMA .....carrier sense multiple access
CSMA/CA.....CSMA with Collision Avoidance
CW .....contention window
CWS-MAC .....Cooperative Wireless Sensor Medium Access Control
D/A.....digital/analog
DAMA .....demand assignment multiple access
DCF.....distributed coordination function
DIFS.....DCF IFS
DM .....Data Message
DRAND .....Distributed RAND
DSMAC .....Dynamic Sensor-MAC
DSP .....digital signal processing
ECN.....explicit contention notification

EIFS .....Extended IFS

es-s .....exactly second-order self-similar

IEEE.....Institute of Electronic and Electrical Engineers

FAMA.....fixed assignment multiple access

FAMA-NTR.....Floor Acquisition Multiple Access – Non-persistent Transmit Request

FLAMA.....Flow-aware Medium Access

FTSP .....Flooding Time Synchronization Protocol

FPRP .....Five-Phase Reservation Protocol

GEO .....geographic earth orbit

GPS .....Global Positioning System

HCF.....hybrid coordination function

HCL.....high contention level

ID .....identification

IFS.....interframe space

iid .....independent, identically distributed

IMA.....Intelligent Medium Access

LCL .....low contention level

LEACH .....Low-energy Adaptive Clustering Hierarchy

LEO.....low earth orbit

LMAC.....Lightweight MAC

LNA .....low noise amplifier

LPL .....low power listening

lrd .....long range dependent

MAC .....medium access control

MACA.....Multiple Access with Collision Avoidance

MACA-BI .....MACA – By Invitation

MACAW.....MACA Wireless

MANET .....mobile ad hoc network

MARCH.....Multiple Access with Reduced Handshake

MCU .....microcontroller unit

MIMO .....multiple-input, multiple-output

NAV.....network allocation vector

NCR .....Neighborhood-aware Contention Resolution

NP .....Neighbor Protocol

NTP.....Network Time Protocol

OPNET®.....Optimized Network Evaluation Tool

PAMAS.....Power Aware Multi-access protocol with Signaling

PC.....personal computer

PCDC .....Power Controlled Dual Channel

PCF .....point coordination function

PCMA .....Power Controlled Medium Access

PHY.....physical

PIFS.....PCF IFS

POWMAC.....Power Control MAC

PRR.....packet reception rate

PAN.....personal area network

PS .....power save

QoS .....quality of service

RAM .....random access memory

RAND .....random

RBS.....Reference Broadcast Synchronization

RCS.....radar cross-section

RISC.....reduced instruction set computer

RTR.....ready-to-receive

RTS/CTS.....request-to-send/clear-to-send

sas-s.....strong asymptotically second-order self-similar

SDMA.....space division multiple access

SECON .....Conference on Sensor and Ad-Hoc Communications and Networks

SEP.....Schedule Exchange Protocol

SIFS.....Short IFS

SINR .....signal-to-interference plus noise ratio

S-MAC.....Sensor-MAC

SNR.....signal-to-noise ratio

srd.....short range dependent

TDMA.....time division multiple access

T-MAC.....Timeout-MAC

TPSN.....Timing-sync Protocol for Sensor Networks

TRAMA .....Traffic-adaptive Medium Access

UAV.....unmanned airborne vehicle

USB.....universal serial bus

UWB .....ultra-wideband

VCO .....voltage-controlled oscillator

WSN.....wireless sensor network

Z-MAC.....Zebra MAC

EXECUTIVE SUMMARY

Researchers have focused on the military application of wireless sensor networks since the introduction of these networks in the late 1990s and it continues to be a topic of significant interest including recent work proposing complex, bandwidth-intensive applications such as unattended battlefield monitoring and enemy signals collection. Unattended battlefield monitoring is a particularly challenging problem because the nodes cannot be replaced or serviced and the communication distance to the collection point can be large. Solutions to this latter problem have been proposed in which the nodes collaborate to perform beamforming to an overhead, unmanned aerial vehicle that is capable of providing a link back to the command and control point. Similarly, recent work in applications such as wireless sensor and actor networks and wireless multimedia sensor networks also considers bandwidth-intensive wireless networks that include sophisticated nodes capable of complex action in response to control input.

While wireless sensor networks have traditionally been comprised of large numbers of small, densely-populated sensor nodes that are constrained in power, processing capability, and memory, these examples of current research represent a group of applications that stretch the capabilities of both the sensor nodes and the underlying network that supports them. In these types of networks, which we term *cooperative wireless sensor networks*, nodes exchange control information to optimize sensing and communication and, therefore, the applications all share a need to provide delay-sensitive inter-node control (packet) communication despite the presence of a large volume of sensor data traffic. The traffic can be divided into distinct flows or groups of flows. The data traffic is typically bandwidth-intensive but is tolerant to individual packet loss because the sensor data are correlated in both time and space. The associated control packet traffic typically requires an end-to-end delay bound and is not tolerant to losses but utilizes significantly less bandwidth. In general, these control packets are smaller and do not arrive as frequently but must be transmitted quickly and reliably.

The research problem addressed in this work, then, is the development of an effective and efficient cooperative wireless sensor network medium access solution that can be applied to these applications. Existing wireless medium access solutions generally fall into two categories: contention-based and scheduled (contention-free). It has been well established that the collision-free approach of scheduled schemes provide high throughput in high demand scenarios at the expense of overhead and packet delay. In comparison, contention-based approaches provide low delay times at low to moderate network loads, but performance begins to degrade rapidly as the load increases and the network becomes saturated. Initial work has been done in the wireless sensor network field to combine the benefits of both approaches in response to changing network load. In these types of approaches, though, medium access is tailored to overall network conditions, not to the characteristics of the individual flow. No medium access scheme has been proposed in literature that is capable of accommodating effectively and efficiently the multiple flows that exist in a cooperative wireless sensor network. While overall network performance may be optimized, individual flows may perform poorly in existing wireless sensor network medium access solutions.

Accordingly, the objective of this research is to achieve the delay performance of contention-based approaches at low demand and the throughput performance of scheduled approaches at high demand on a *per flow* basis. The fundamental contribution of this dissertation is the introduction of *traffic-adaptive, flow-specific medium access* and the development of a novel energy-efficient, traffic-adaptive, flow-specific, medium access scheme. By this, we mean that the medium access scheme is capable of concurrently providing different medium access service to different traffic flows (i.e., on a per flow basis) and dynamically switching flows between multiple medium access service types to respond to traffic variations. Our hypothesis is that by adapting to both the traffic flow patterns of the individual flows and the contention levels within the medium as a whole, the medium access scheme will be able to provide better throughput and delay performance across the aggregate network and will be capable of addressing flow-specific delay, throughput and reliability requirements.

This research effort began by examining the traffic within wireless sensor network applications and we postulated that throughput and delay performance could be improved over existing medium access solutions by providing medium access service on a per flow basis. We introduced the novel concepts of flow-specific medium access and traffic-adaptive, flow-specific medium access and, based on these ideas, we proposed a traffic-adaptive, flow-specific medium access scheme. After conducting a thorough performance analysis, along with accompanying simulation, we turned our attention towards the energy efficiency of the scheme and proposed an energy-efficient version designed to meet the constraints of a wireless sensor network implementation. We also proposed the use of a modified version of the proposed scheme to provide medium access for a networked satellite system. A number of significant contributions have emanated from this research effort.

This research identifies and characterizes multiple and distinct flows within a wireless sensor network and introduces the term *cooperative wireless sensor network*. In a cooperative wireless sensor network, sensors exchange information to coordinate efforts and maximize application-related performance. This work specifically identifies a high-demand, loss-tolerant flow associated with the sensor data traffic and a low-demand, loss-intolerant, delay-sensitive flow associated with the sensor control traffic. We assert, therefore, that a medium access solution for wireless sensor networks should provide a high throughput medium access service to support the former and a reliable service with minimum end-to-end delay to support the latter.

This dissertation, then, formally develops *traffic-adaptive, flow-specific medium access*, a groundbreaking medium access technique that provides medium access on a per flow basis rather than in aggregation, and proves that it is capable of providing better delay performance than contention, non-contention, and hybrid approaches. To realize this performance advantage, this proposes traffic-adaptive Cooperative Wireless Sensor Medium Access Control (CWS-MAC), a novel traffic-adaptive, flow-specific medium access scheme capable of providing contention- or non-contention-based medium access service on a per flow basis. Traffic-adaptive CWS-MAC is shown to outperform both slotted, non-persistent CSMA (upon which the IEEE 802.11 standard is based) and

TDMA. In the accompanying analysis, this research develops a general model for traffic-adaptive, flow-specific medium access control and demonstrates that traditional contention, non-contention, and hybrid medium access schemes are special cases of this model. In conjunction with this analysis, to our knowledge, this work also develops the first published throughput and delay performance analysis for slotted ALOHA with periodic server vacations.

This dissertation also proposes an energy-efficient version of traffic-adaptive CWS-MAC that employs an adaptive sleep cycle coordinated through the use of preamble sampling and is capable of achieving low duty cycles required in current wireless sensor network applications. A preamble sampling probability parameter is introduced to manage the trade-off between energy efficiency and network throughput and delay performance. To our knowledge, this work also provides the first comprehensive energy efficiency comparison between centralized and distributed solutions in wireless networks. The analysis includes energy consumption in the transmit, receive, idle and sleep states. A performance threshold is shown to exist between these approaches, which can be exploited through the use of preamble sampling. Finally, as a novel capstone application, this dissertation proposes a flow-specific medium access technique to accommodate the large and variable propagation delays and dynamic traffic requirements in a networked satellite system.

This research represents the pioneering work in traffic-adaptive, flow-specific medium access. A number of exciting research problems come out of this effort. Of particular interest among these is the most effective and efficient implementation of both the contention and non-contention modes, as well as the application of traffic-adaptive, flow-specific medium access to both established and unconventional wireless network applications. Opportunities for future research along these lines also include testbed and field results for traffic-adaptive CWS-MAC.

ACKNOWLEDGMENTS

I am grateful for the funding support provided for my research by U.S. Special Operations Command.

I would like to express my sincere gratitude and appreciation to my advisor, Dr. Murali Tummala. His mentorship, guidance and friendship across 17 years have been invaluable in all of my postgraduate research and academic accomplishments. This dissertation is a reflection of the high standard he sets and inspires others to achieve.

I would like to thank my co-advisor, Dr. John McEachen, whose consistent support of my research provided many important opportunites to present this work and receive feedback from peers within the field.

I am also indebted to the other members of my dissertation committee—Dr. Hersch Loomis, Dr. Roberto Cristi, and, particularly, Dr. Bret Michael—whose keen insights, recommendations, and encouragement to publish were instrumental in the quality of this work.

Finally, I would like to thank my wonderful wife, Rebecca, and our beautiful daughters, Haley and Victoria, for their love, patience, encouragement, and support—it has been a tremendous blessing and I truly appreciate it.

THIS PAGE INTENTIONALLY LEFT BLANK

I. INTRODUCTION

Researchers have focused on the military application of wireless sensor networks since the introduction of these networks in the late 1990s and it continues to be a topic of significant interest including recent work proposing complex, bandwidth-intensive applications such as unattended battlefield monitoring [1] and enemy signals collection [2],[3]. Unattended battlefield monitoring is a particularly challenging problem because the nodes cannot be replaced or serviced and the communication distance to the collection point can be large. The authors of [4] propose a solution to this latter problem in which the nodes collaborate to perform beamforming to an overhead UAV that is capable of providing a link back to the command and control point. Similarly, recent work in applications such as wireless sensor and actor networks [5] and wireless multimedia sensor networks [6] also considers bandwidth-intensive wireless networks that include sophisticated nodes capable of complex action in response to control input.

A number of cooperative distributed radar approaches have also been explored in literature within the last five years. Multiple-input multiple-output (MIMO) radars [7],[8],[9] are designed to achieve spatial diversity gains and are motivated by the related work in MIMO communication theory [10]. For example, statistical MIMO radar [7],[11] mitigates target radar cross-section (RCS) fading by transmitting orthogonal waveforms that are sufficiently separated in distance to ensure the reflected signals are uncorrelated. Collaborative beamforming-based approaches such as that described above can be seen as distributed phased-array radars that seek to improve the received signal-to-noise ratio (SNR) by correlating the transmitted waveforms [12],[13],[14]. These approaches require the calculation and distribution of a set of complex weights as well as phase and frequency synchronization. The netted radar work of [15] also requires synchronization among the transmitters and receivers to realize its theoretical SNR gain of the square of the number of participating radar nodes.

While wireless sensor networks have traditionally been comprised of large numbers of small, densely-populated sensor nodes that are constrained in power,

processing capability, and memory [16],[17], these examples of current research, though, represent a group of applications that stretch the capabilities of both the sensor nodes and the underlying network that supports them. Of particular interest, nodes exchange control information to optimize sensing and communication and, therefore, the applications all share a need to provide delay-sensitive inter-node control (packet) communication despite the presence of a large volume of sensor data traffic. For example, in a wireless multimedia application, video camera-equipped nodes transmit high-bandwidth streaming video while camera control inputs (e.g., camera movement and lens focus) must be injected into the network traffic to optimize sensor node and resource utilization [6]. Similarly, in the case of distributed beamforming, the transmission of complex weights to the participating nodes to form the beam competes directly with the high-bandwidth sensor data transmitted through the beam [13]. In these types of networks, which we term *cooperative wireless sensor networks* [18], traffic can be divided into distinct flows or groups of flows. The data traffic is typically bandwidth-intensive but is tolerant to individual packet loss because the sensor data are correlated in both time and space. The associated control packet traffic typically requires an end-to-end delay bound and is not tolerant to losses but utilizes significantly less bandwidth. In general, for this traffic class, the packets are smaller and do not arrive as frequently but must be transmitted quickly and reliably.

A. OBJECTIVE

The research problem addressed in this work is the development of an effective and efficient cooperative wireless sensor network medium access solution that can be applied to these applications. Existing wireless medium access solutions generally fall into two categories: contention-based and scheduled (contention-free). It has been well established that the collision-free approach of scheduled schemes, such as [19], provide high throughput in high demand scenarios at the expense of overhead and packet delay. In comparison, contention-based approaches, such as [20],[21],[22] provide low delay times at low to moderate network loads, but performance begins to degrade rapidly as the load increases and the network becomes saturated. Initial work has been done in the

wireless sensor network field to combine the benefits of both approaches in response to changing network load. Most notably, the authors of [23] provide a contention-based approach that utilizes TDMA framing to provide “hints” for contention resolution. In these types of approaches, though, medium access is tailored to overall network conditions, not to the characteristics of the individual flow. No medium access scheme has been proposed in literature that is capable of accommodating effectively and efficiently the multiple flows that exist in a cooperative wireless sensor network. While overall network performance may be optimized, individual flows may perform poorly in existing wireless sensor network medium access control solutions.

Accordingly, the objective of this research is to achieve the delay performance of contention-based approaches at low demand and the throughput performance of scheduled approaches at high demand on a *per flow* basis. The fundamental contribution of this dissertation is the introduction of *traffic-adaptive, flow-specific medium access* and the development of a novel energy-efficient, traffic-adaptive, flow-specific, medium access scheme. By this, we mean that the medium access scheme is capable of concurrently providing different medium access service to different traffic flows (i.e., on a per flow basis) and dynamically switching flows between multiple medium access service types to respond to traffic variations. Our hypothesis is that by adapting to both the traffic flow patterns of the individual flows and the contention levels within the medium as a whole, the medium access scheme will be able to provide better throughput and delay performance across the aggregate network and will be capable of addressing flow-specific delay, throughput and reliability requirements. By necessity, this scheme must be both flow aware as well as traffic aware, and can be considered a cross-layer approach involving the application, link, and physical layer.

This work develops both the flow-specific medium access scheme and the traffic-adaptive mechanism to allow it to respond to changes in traffic demand. It also investigates the energy-efficiency of the solution and proposes an adaptive sleep and wake cycle utilizing preamble sampling. Although the proposed solution includes a TDMA-based non-contention mode, this dissertation does not address the slot assignment or slot synchronization algorithms. A number of centralized and distributed solutions to

these problems have been proposed in the literature and are discussed in Chapter II. Additionally, this effort leaves for future research the formal analysis of the underlying medium access algorithm, which should include an evaluation of the safety, liveness, and fairness of the proposed scheme.

B. RELATED WORK

Medium access control for wireless networks and wireless sensor networks has been a well-studied problem and a number of solutions have been proposed in the literature. The wireless sensor network schemes are typically based on the earlier wireless techniques and are highlighted by energy efficiency improvements. This section briefly outlines the most relevant of these and discusses the medium access approach proposed in this dissertation relative to this existing work. More detailed coverage of these protocols can be found in Chapter II.

Carrier sense multiple access (CSMA) [21] and its many variants are among some of the most widely studied proposals to provide contention-based medium access to wireless networks. They feature good throughput and delay performance at low to medium loads, but performance falls off rapidly as the load increases. Furthermore, the protocols are very energy-intensive and require nodes to constantly monitor the channel. The IEEE 802.11 standard [24] is built upon the slotted, non-persistent CMSA protocol and reflects these benefits and shortcomings. The medium access scheme proposed in this dissertation takes advantage of the contention-based performance at low-medium traffic demand, but improves upon the high demand performance by switching away from a contention-based approach as performance degrades. This research effort also improves the energy efficiency of the contention-based mode by implementing a sleep and wake cycle through the use of preamble sampling.

The IEEE 802.15.4 standard [25] and Sensor-MAC (S-MAC) [22] are the most widely studied among a large group of contention-based medium access schemes that have been proposed for wireless sensor networks. IEEE 802.15.4 [25] uses both slotted and unslotted CSMA and features a sleep cycle that improves energy efficiency. A contention-based solution, it reflects the same performance trends common to other

CSMA-based approaches. Furthermore, it relies on a centralized controller to implement the sleep cycle. S-MAC [22] provides a distributed solution to sleep cycle coordination, but still suffers from the fall off in performance at heavy loads. Again, the scheme proposed in this work improves performance at heavy demand over contention-based approaches while providing a distributed solution to the sleep cycle coordination.

Time division multiple access (TDMA) [26] is a common contention-free approach to medium access. It performs well at high loads, but suffers from increased overhead and inefficient operation at low to medium loads. The Traffic-adaptive Medium Access (TRAMA) protocol [27] is a well-studied TDMA-based solution for wireless sensor networks that allows nodes to sleep through the use of a preschedule scheme. The scheme proposed in this dissertation incorporates the performance benefits of TDMA at high traffic demand, but switches to a contention-based scheme at low to medium loads to improve on the performance of TDMA. It also provides a preamble sampling mechanism to improve on the energy efficiency of TDMA.

Finally, Z-MAC [28] is the most well known of several hybrid approaches that have been proposed in the literature for wireless sensor networks. As in this dissertation, these hybrid solutions attempt to combine the performance of contention-based medium access approaches at low-to-moderate loads and non-contention-based approaches at high loads. Z-MAC uses a CSMA-based medium access scheme in the contention mode and a TDMA-based scheme in the non-contention mode. Energy savings are achieved utilizing a sleep schedule that is coordinated through preamble sampling. In many aspects, the work here is similar to Z-MAC. The salient difference is that the novel work presented in this dissertation treats the flows *individually* while Z-MAC and other existing hybrid solutions treat the flows in aggregation. This important point, discussed in detail in Chapter III, underlies the research effort of this dissertation and is chiefly responsible for the throughput and delay performance improvements achieved.

C. ORGANIZATION

This dissertation is organized as follows. Chapter II provides a background overview of relevant topics in wireless sensor networks and a discussion of medium

access control considerations. It also provides a brief survey of existing solutions in both wireless medium access and wireless sensor network medium access.

Chapter III formally introduces the novel concept of traffic-adaptive, flow-specific medium access and provides an example to contrast its operation relative to contention, non-contention and hybrid approaches. This chapter also compares the delay performance of traffic-adaptive, flow-specific medium access to these traditional solutions and formally shows that it is capable of outperforming them.

Chapter IV proposes the novel traffic adaptive, flow-specific medium access scheme. Delay and throughput performance analysis is provided along with supporting simulation. This analysis includes the introduction of a general traffic-adaptive, flow-specific performance model (of which contention, non-contention, and hybrid medium access are shown to be special cases) and, to our knowledge, the first published performance analysis for slotted ALOHA with periodic server vacations.

The energy efficiency of the proposed medium access scheme is the focus of Chapter V, which begins by providing the first comprehensive comparison of the energy efficiency of centralized and distributed solutions. From this, the role of preamble sampling on energy efficient is identified and applied to our proposed scheme. The chapter includes detailed energy efficiency and duty cycle analysis along with supporting simulation results.

Chapter VI provides a capstone application of the work, which proposes a flow-specific medium access solution for a networked satellite system. The chapter includes analysis of the effect of the large propagation distance on medium access control performance and provides relevant simulation results.

In conclusion, Chapter VII provides a summary of our research and a listing of the significant contributions as well as some suggestions for future research that can build upon this work. An appendix is provided that includes the salient OPNET® simulation code and models.

II. BACKGROUND

This chapter reviews the body of existing work that forms the foundation of the research effort described in this dissertation. Specifically, it provides an overview of relevant topics in wireless sensor networks, a discussion of medium access control considerations and a brief survey of existing solutions in both wireless medium access and wireless sensor network medium access. The intent of this chapter is to provide the context necessary to understand fully both the research problem and the proposed solution. The topics presented here are referenced, both implicitly and explicitly, throughout the remainder of the dissertation.

A. WIRELESS SENSOR NETWORKS

The chapter begins with an overview of wireless sensor networks to give the reader an understanding of the constraints and design considerations faced by wireless sensor network medium access control researchers. This section discusses the networks themselves as well as the characterization, performance analysis, and estimation of traffic within these networks.

1. Wireless Sensor Network Technology

This first section provides a model for a wireless sensor node and discusses characteristics of a wireless sensor network. It concludes the section by providing examples of the current generation of wireless sensor node technology.

a. Wireless Sensor Node Model

Designed to be small in size, wireless sensor nodes are limited in power, processing, storage, bandwidth, and range capability [16]. A schematic of a typical sensor node is provided in Figure 1, which illustrates the suite of onboard sensors (with accompanying A/D convertors), the processor, the transceiver, the memory storage, and the battery.

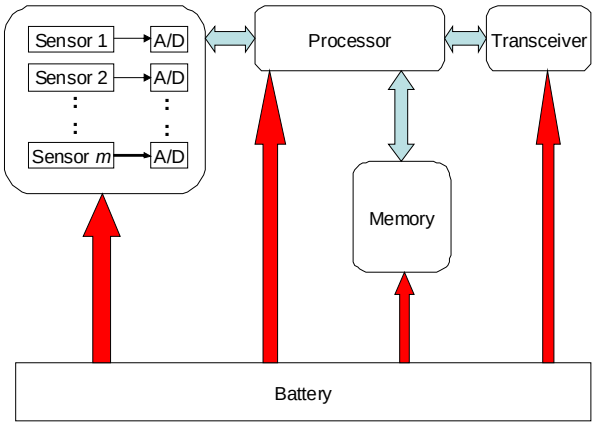


Figure 1. A schematic of a typical sensor node (After [29]).

Due to the battery-powered nature of the motes and the unattended applications, energy conservation to extend network lifetime is a primary objective in the design of wireless sensor networks [30]. The original Mica2 motes consume 720 nJ/bit transmit and 4 nJ/operation [31] and it is important to recognize the fundamental characteristic that communication tends to be more costly in terms of energy consumption than processing. This will often lead designers to trade onboard processing for communication. Another characteristic of sensor nodes is their capability to transition into a reduced power "sleep" state when not in use. Important considerations in sensor node design include how quickly the node can transition between the sleep, wakeup, and active states, how much energy it expends, and how quickly it can get work done in the active state.

b. Wireless Sensor Networks

A wireless sensor network is a collection of tens to thousands of wirelessly connected and self-organizing sensor nodes [32]. They are designed to provide "up close" (embedded) sensing as well as redundancy and improved accuracy. Applications of wireless sensor networks include environmental, bio-medical, seismic, and structural monitoring; industrial automation; and military detection and surveillance.

Similarities between a mobile ad hoc network (MANET) and a wireless sensor network (WSN) include the ad hoc topology, the shared (broadcast) communication medium, and the physical-layer connectivity issues [17]. Wireless sensor

networks, however, are also constrained in power, processing, storage, bandwidth, and range [16],[32]. The limited range is tied to the limited power and, as a result, WSNs tend to be dense and scalability is an issue (both in terms of the size of the network and the size of an individual node) [17]. WSNs tend to experience frequent topology changes and nodes are often not assigned global IDs due to the scale of the network [16]. WSNs are typically application-specific and, to date, most WSN research has assumed limited or no mobility [17]. The traffic is driven by the sensed data and accompanying queries and can often be characterized a priori as opposed to MANET traffic, which is highly interactive [17]. Additionally, this traffic tends to be correlated in both time and space because of the close proximity and redundancy of the sensor nodes [33]. Finally, the traffic flow can often be seen to be "gathercast" [17], meaning that it flows from many sources to one destination.

Wireless sensor networks are application dependent and, not surprisingly, the generated traffic is also application dependent. Application traffic generation from the sources to the sink(s) in sensor network applications can be classified as either time-driven or event-driven. Examples of the former include periodic reporting to support environmental monitoring while an example of the latter is event-based reporting to support intrusion detection.

c. Specific Wireless Sensor Motes

This section provides an overview of two current-generation wireless sensor nodes. The first is the TELOSB mote [34], the latest in the long line of Berkeley motes (now produced by Crossbow, Inc.). The second is a relative newcomer, the SunSPOT [35], a Java-based sensor mote from Sun Microsystems.

Available Crossbow sensor node products include the MICA2 mote [36], the MICAz mote (a MICA2 mote with an improved microcontroller and an IEEE 802.15.4 radio) [37] and the TELOSB mote [34], all descendents of the original Berkeley MICA mote [38]. Shown in Figure 2, the TELOSB mote processor and radio board include a T1 MSP430 microcontroller with 10kB RAM, a 16-bit RISC processor with 48K Program Flash, a IEEE 802.15.4 radio at 250 kbps and 1 MB external data flash. The Telos mote runs the Tiny OS operating system (1.1.11 or higher) and is powered by two

AA batteries (or a connected USB port). Onboard integrated sensors include temperature, light and humidity [34]. The MICAz mote nominally consumes less than 15 μA in the sleep state and 8 mA in the active state [37]. It is capable of transitioning from the sleep to the active state in 60 μs and contains an 8 MHz 8-bit processor. In contrast, the TELOSB mote nominally consumes 5.1 μA in the sleep state and 1.8 mA in the active state [34]. It is capable of transitioning from the sleep to the active state in 290 ns and contains a 4-8 MHz 16-bit processor.

Figure 2. Telos sensor mote (From [34]).

The SunSPOT wireless sensor mote [35] is a recent release from Sun Microsystems that is designed to foster wireless sensor network research by providing an open, Java-based platform that runs on bare metal vice a vendor-specific operating system. Shown in Figure 3, the SunSPOT includes a 180 MHz 32-bit ARM920T processor with 512KB RAM and 4 MB Flash memory. Communications are provided by a 2.4 GHz IEEE 802.15.4 compliant radio with an integrated antenna. The power source is a 3.7V, rechargeable, 720 mAh lithium-ion battery and the mote draws 32 uA in deep sleep mode. With both the CPU and transceiver powered, node lifetime is approximately seven hours while lifetime in the deep sleep mode can be as long as 900 days. The sensor board includes temperature and light sensors, an accelerometer, and a set of six analog inputs that are fed into an analog-to-digital converter. A USB interface is included to provide connectivity to a local PC.

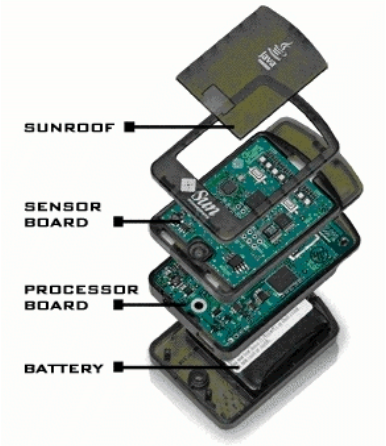


Figure 3. SunSPOT sensor mote (From [35]).

2. Traffic Characterization, Estimation and Performance Modeling

This section examines the work that has been done to date with respect to characterizing and measuring traffic flows within a sensor network. It begins with a discussion of existing traffic characterization studies for wireless sensor networks and then ties these to performance analysis using the appropriate queuing models. The section closes by identifying proposed mechanisms to measure and estimate traffic flows.

a. Traffic Characterization: Poisson versus Self-similar

A common assumption in traditional queuing analysis (originally based on observations of voice traffic in a switched telephone network [39]) is that the arrivals are Poisson distributed

$$\Pr[X = k] = \frac{\lambda^k}{k!} e^{-k} \quad \text{for } k = 0,1,2... \text{ and } \lambda > 0 \tag{1}$$

with a mean arrival rate of λ. Therefore, the interarrival times are independent and identically distributed (iid) with an exponential distribution

$$f(x) = \begin{cases} 0 & \text{for } x < 0 \\ \lambda e^{-\lambda x} & \text{for } x \geq 0 \end{cases}$$
$$F(x) = \begin{cases} 0 & \text{for } x < 0 \\ 1 - e^{-\lambda x} & \text{for } x \geq 0 \end{cases} \tag{2}$$

and a mean interarrival time of 1/λ. A renewal process is a generalization of the Poisson process and is simply defined as a process in which the interarrival times are iid [40].

In contrast to a Poisson process, which smoothes out as the scale becomes larger, self-similarity can be defined as "scale invariance" where an object appears identical at any level of magnification [41]. A real process $Y(t)$ is self-similar with self-similarity parameter H (called the Hurst parameter), if all finite dimensional distributions of $Y(t)$ are identical to the finite dimensional distributions of $a^{-H}Y(at)$ for all $a > 0$ and $0 < H < 1$ [42]. In other words, the original process $Y(t)$ and all of its time-scaled versions, $Y(at)$, share the same distribution in all dimensions provided that the magnitude of $Y(at)$ is normalized by a^{-H}. A discrete wide-sense stationary random process X can be said to be exactly second-order self-similar (es-s) with $H = 1 - \frac{\beta}{2}$ $(0 < \beta < 1)$ if [43]

$$\left.\begin{array}{l} \mathrm{var}\left(X^{(m)}\right) = \dfrac{1}{m^{\beta}}\mathrm{var}\left(X\right) \\ \text{and} \\ r_{X^{(m)}}\left(k\right) = r_X\left(k\right) \end{array}\right\} \quad \text{for all } m = 1, 2, 3 \ldots \text{ and } k \geq 0 \tag{3}$$

where $\mathrm{var}\left(X\right)$ is the variance of X, $r_X(k)$ is the autocorrelation function of X (defined as the autocovariance of X normalized by the variance) and $X^{(m)}$ is the time aggregated version of X created by averaging X over non-overlapping blocks of size m. A discrete random process X is said to be asymptotically second-order self-similar (as-s) if

$$\lim_{m \to \infty} r_{X^{(m)}}\left(k\right) = r_X\left(k\right). \tag{4}$$

In [44], the authors point out that (3) contains some redundancy and offer the following revised definition of a self-similar process. A discrete random process X is exactly second-order self-similar if

$$r_X(k) = \frac{1}{2}\left[\left(k+1\right)^{2-\beta} - 2k^{2-\beta} + \left(k-1\right)^{2-\beta}\right] \tag{5}$$

and asymptotically second-order self-similar if

$$\lim_{m\to\infty} r_{X^{(m)}}(k) = \frac{1}{2}\left[(k+1)^{2-\beta} - 2k^{2-\beta} + (k-1)^{2-\beta}\right]. \quad (6)$$

A strong asymptotically second-order self-similar (sas-s) process is then defined [45] as one in which

$$\lim_{m\to\infty} \mathrm{var}\left(X^{(m)}\right) \sim m^{-\beta} \quad (7)$$

where ~ indicates that expressions are asymptotically proportional (i.e., their ratio tends to a constant in the limit).

The degree of self-similarity (equivalently, the value of the Hurst parameter, *H*) can be estimated using one of three approaches: (1) analysis of the variance-time plot of the processes $X^{(m)}$, (2) analysis of the R/S plot of *X*, or (3) computation of the Whittle's estimator from the spectral density [43]. Although the latter can be considered the most refined approach, the former is most often used in practice (e.g. [46],[47]). The variance-time plot is derived from (3) by taking the log of both sides to arrive at

$$\log\left(\mathrm{var}\left(X^{(m)}\right)\right) \sim \log\left(\mathrm{var}\left(X\right)\right) - \beta \log m \quad (8)$$

for large values of *m*. The value of β (and subsequently *H*) can then be estimated by plotting $\mathrm{var}\left(X^{(m)}\right)$ versus *m* on a log-log graph and measuring the slope of the resulting line.

A few studies have been conducted in an attempt to characterize the overall traffic encountered in a wireless sensor network. In [47], the author contends that ad hoc wireless traffic is self-similar in nature, but these findings are based on a variance-time plot of a single mpeg application observed for 12.5 minutes. More recently, the authors of [46] also uses variance-time plots to conclude that the packet length for a one-hop direct topology and both the packet length and the packet interarrival time for a multi-hop daisy-chained topology are not self-similar. Additionally, although the packet interarrival time for the daisy-chained topology was found to be self-similar, it had a Hurst parameter of only 0.60015. These results were found by collecting more than 2.1 million packets for a network of six motes over a period of 180 hours. In this study, the network was modeled as time-driven with periodic reporting intervals. To the best of our

knowledge, no studies have been conducted to determine whether event-driven WSN traffic is self-similar. The work presented here assumes the traffic is Poisson and the performance analysis included reflects the queuing models outlined in the next section. We remain cognizant of this assumption and leave it to further research to explore the potential impact of self-similar traffic on the performance of the proposed scheme.

b. Performance Analysis using Queuing Models

Performance analysis of communication networks can be achieved by modeling them as systems of queues. The classic communication queuing theory reference is [48]. The summary provided in this section is based on this reference as well as the discussions in [49] and [41]. A basic single server queuing system and accompanying notation are provided in Figure 4. Little's formula provides a fundamental relationship between arrival rate λ, service time T_s, and number of customers r in the system and is given by

$$E[r] = \lambda E[T_s]. \tag{9}$$

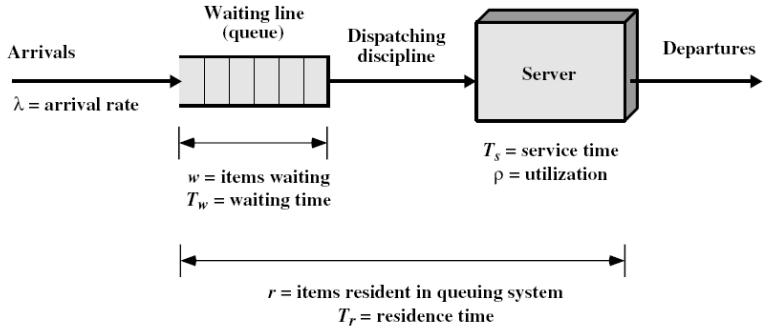
From this formula, equations for the remaining parameters in Figure 4 can be developed.

As noted above, this research effort assumes that the traffic is Poisson-distributed. Thus, when both the interarrival times and the service times are exponential (M/M/n in Kendall's notation), the system can be modeled and analyzed as a Markov chain because exponential random variables possess the memoryless property. Systems with generally distributed service times (M/G/n) can be analyzed through the use of the Pollaczek-Khinchin mean value formula [48]

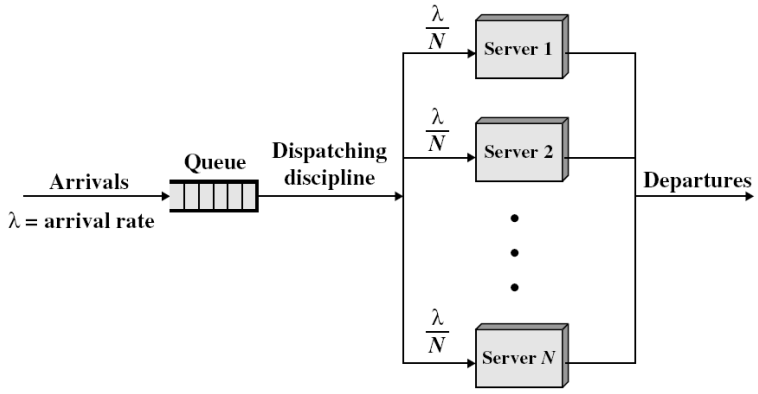
$$T_w = \frac{\lambda\left(\sigma_{T_s}^2 + T_s^2\right)}{2\left(1-\lambda T_s\right)} \tag{10}$$

where we see that the waiting time is a function of only the second-order statistics (mean service time T_s and variation $\sigma_{T_s}^2$) of the service time distribution. The ratio of $\frac{\sigma_{T_s}}{T_s}$ can be viewed as a normalized metric of the variability in the process and is known as the coefficient of variation. A value of one equates to an exponential distribution (M/M/n) while values less than one suggest the performance will be better than the exponential

case and values greater than one indicate that the M/G/n model is appropriate. A constant distribution will have a coefficient of variation of zero. A listing of some of the important (and well-known) formulas for single sever, multiserver, and priority queues can be found in [41].



(a)



(b)

Figure 4. Basic (a) single server and (b) multiserver queuing systems with accompanying notation (From [41]).

Jackson's theorem [48] allows the application of the principles of traffic partitioning, merging and tandem queues (as shown in Figure 5) to a network of queues by establishing that each node within the network is an independent, M/M/n queuing system provided that the arrivals are Poisson, the service times are independent, exponential random variables and that a packet immediately proceeds to the next node (or exits the system) after being served. As we shall see in the following, some work has

been done to develop similar principles for traffic flows where the arrivals are self-similar vice Poisson. It should also be noted that although the service times are not strictly independent in a network of queues (the packet length remains the same as a packet traverses the network), it is a reasonable approximation [48].

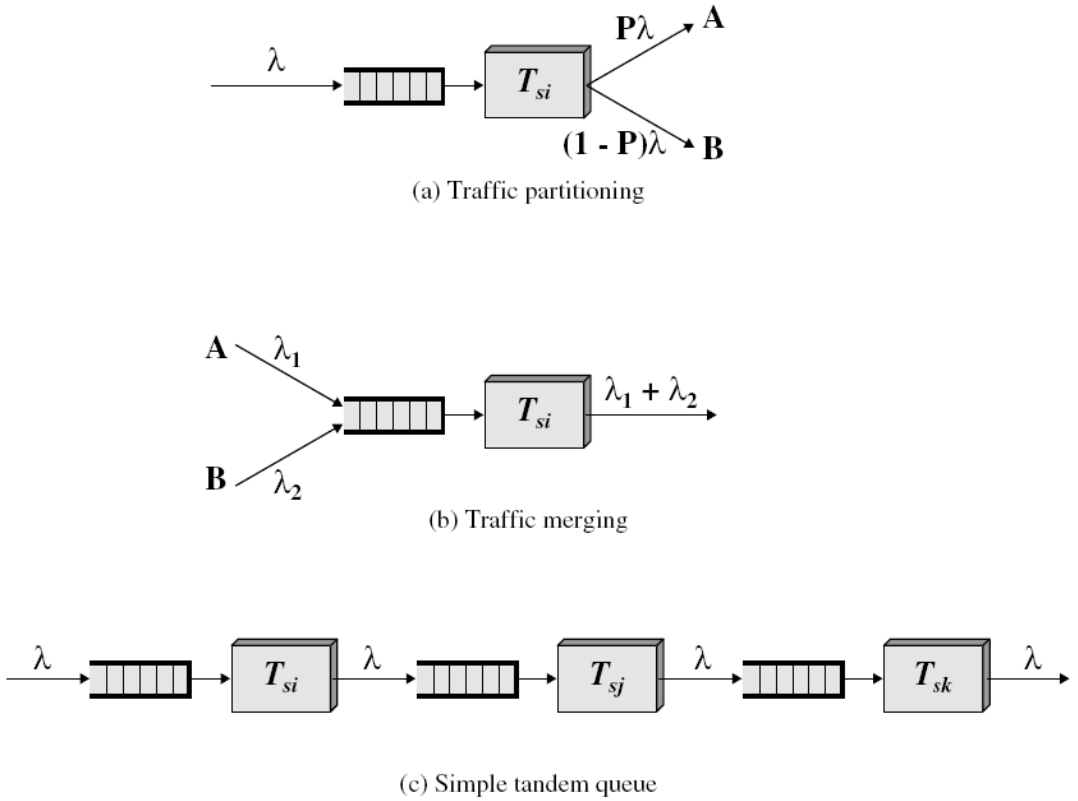


Figure 5. Principles of traffic partitioning, merging and tandem queues for systems with Poisson arrivals and independent, exponentially distributed service times (From [41]).

While the analysis in this dissertation assumes Poisson-distributed traffic, the effect of self-similarity on network performance is reflected in the following queue size analysis to provide a basis upon which to explore potential follow-on research. It has been shown [50] that for self-similar traffic, the queue size, *q*, as a function of the utilization, ρ, is

$$q = \frac{\rho^{\frac{1}{2(1-H)}}}{(1-\rho)^{\frac{H}{(1-H)}}}. \tag{11}$$

Substituting 0.5 for *H*, this reduces to the queue size result for classical, Poisson-based queuing analysis of

$$q = \frac{\rho}{(1-\rho)} \tag{12}$$

and, as can be seen in Figure 6, which plots queue size versus utilization for self-similar, M/M/1, and M/D/1 systems, the queue sizes will, in general, be larger in the case of self-similar traffic. The effect of merging self-similar streams was first explored in [44] and then more fully developed in [45]. The results are shown Table 1. In [45], the authors also proved that, assuming no packet drops (i.e., infinite queue length), if the queue length has a finite second order distribution then an input process that is exactly second-order self-similar with Hurst parameter *H* will produce an output process that is also exactly second-order self-similar with Hurst parameter *H*. Summarizing the findings in [45], the merging of two self-similar streams results in a stream that is also self-similar with the same characteristics and a "bounded" server does not alter the self-similar characteristics of a stream.

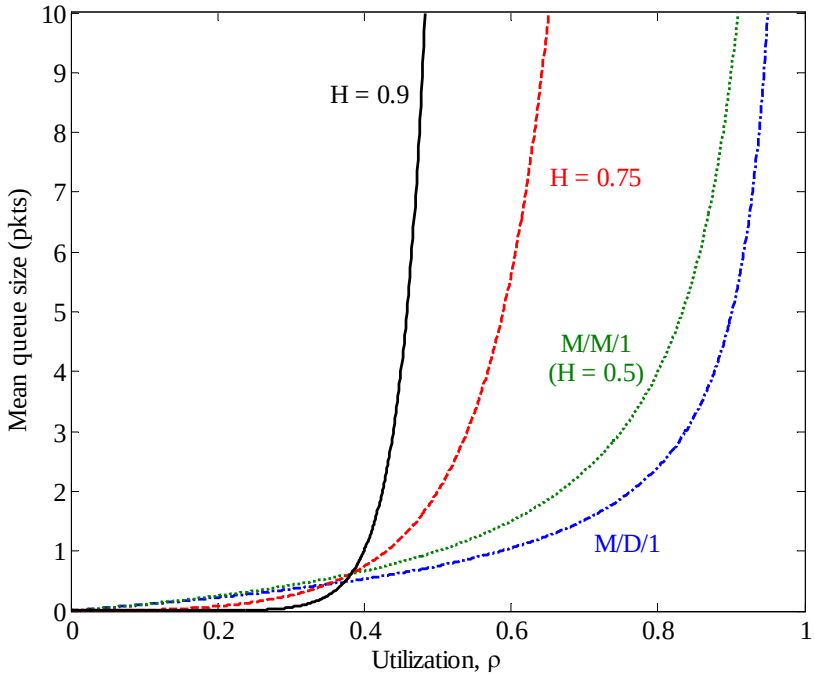


Figure 6. Queue size as a function of utilization under different traffic models (After [41]).

THIS PAGE INTENTIONALLY LEFT BLANKInput Stream, X_1	Input Stream, X_2	Output Stream, $Y = X_1 + X_2$	Conditions
lrd with H_1	lrd with H_2	lrd with $H = \min(H_1, H_2)$	X_1 and X_2 are uncorrelated
sas-s with H_1	sas-s with H_2	sas-s with $H = \min(H_1, H_2)$	X_1 and X_2 are uncorrelated
lrd with H_1	lrd with H_2	lrd with $H = \min(H_1, H_2)$	X_1 and X_2 are correlated and there exists $\beta_3, \beta_4 > \min(\beta_1, \beta_2)$ such that $\begin{cases} \lim_{k\to\infty} \lvert C(X_1(t), X_2(t+k)) \rvert \le c_3 k^{-\beta_3} \\ \text{and} \\ \lim_{k\to\infty} \lvert C(X_1(t+k), X_2(t)) \rvert \le c_4 k^{-\beta_4} \end{cases}$
sas-s with H_1	sas-s with H_2	sas-s with $H = \min(H_1, H_2)$	X_1 and X_2 are correlated and there exists $\beta_3, \beta_4 > \min(\beta_1, \beta_2)$ such that $\begin{cases} \lim_{k\to\infty} \lvert C(X_1(t), X_2(t+k)) \rvert \le c_3 k^{-\beta_3} \\ \text{and} \\ \lim_{k\to\infty} \lvert C(X_1(t+k), X_2(t)) \rvert \le c_4 k^{-\beta_4} \end{cases}$
lrd with H	srd	lrd with H	X_1 and X_2 are uncorrelated second-order stationary processes and X_2 satisfies $r_{X_2}(k) \sim k^{-\beta_2}$ where $1 \le \beta_2 \le 2$
sas-s with H	srd	sas-s with H	X_1 and X_2 are uncorrelated second-order stationary processes and X_2 satisfies $\lim_{m\to\infty} \text{var}\left(X_2^{(m)}\right) \sim m^{-1}$

Table 1. Characteristics of the result of the merger of self-similar streams.

c. Traffic Estimation

Traffic estimation techniques are often closely tied to research in both congestion control and traffic-adaptive protocols. The three common approaches to traffic estimation are channel sampling, queue occupancy measurements, and packet arrival/service time measurements [51],[52].

In channel sampling schemes, such as [53], the noise level of the medium is sampled periodically and a utilization factor is calculated. The authors in [23] propose an indirect and passive approach to channel sampling where the number of backoffs due to a busy medium is averaged to determine the level of congestion vice actually sampling the medium. The primary disadvantage of these channel sampling solutions is that they are only capable of measuring the aggregate traffic load and, hence, do not provide flow-specific estimation.

Queue occupancy techniques involve direct measurement of queue size and have the advantage that they can be directly applied to multiple flow schemes by considering flow-specific queues individually. The congestion control mechanisms proposed in [51], [54], and [52], and the traffic adaptive medium access scheme in [19] all make use of queue occupancy measurements to estimate current traffic load. It should also be noted that contention-based medium access solutions, in general, possess an implicit queue occupancy-based mechanism in that the nodes only contend for the medium when their queue size is non-zero. The authors in [51] demonstrate that congestion detection utilizing queue occupancy performs as well or better than channel sampling and point out that it is easier to implement. Packet arrival and service rate approaches [55],[56],[57] are a variation of queue occupancy in which nodes directly measure arrival rates and service times (typically using exponentially weighted moving averages) vice queue size to arrive at an estimated load per node.

B. MEDIUM ACCESS CONSIDERATIONS

This section reviews the current research in a number of areas that directly relate to the development of an energy-efficient, flow-specific medium access control scheme.

It begins by modeling the wireless channel and then moves to challenges that are specific to either contention-based or contention-free access. It then provides a survey of collision avoidance techniques to support contention-based solutions and distributed slot assignment and time synchronization proposals to support TDMA solutions. The section concludes with a discussion of MAC layer power management approaches designed to address the power constraints in both MANETs and WSNs and a brief overview of cross-layer design.

1. Modeling the Wireless Communication

This section discusses the wireless communication models that have been proposed in literature. It includes both the link quality model, which is designed to capture the RF propagation and reception, and the interference model, which models the probability of correct packet reception given interference from competing transmissions.

a. Link Quality Model

Many protocols implicitly assume the spherical, path loss RF propagation model given by [58]

$$P_r = P_t\left(\frac{\lambda}{4\pi d}\right)^{\alpha} \tag{13}$$

or, equivalently in dB,

$$L_p = 10\log P_t - 10\log P_r = 10\alpha\log\left(\frac{4\pi d}{\lambda}\right) \tag{14}$$

where L_p (in dB) is the path loss, P_t is the transmitted power, P_r is the received power, and d is the distance between the transmitter and the receiver. This model reflects the observation that the average received SNR decreases logarithmically with distance [59]. The path loss exponent α is an indication of how fast the signal power drops off as a function of distance. The minimum value of two represents the ideal case of free space while smaller values model rural areas and larger values model urban areas [58].

In practice, though, received radio signal strength is not spherical as a function of distance due to the effect of the surrounding environment [59]. The difference between the average, spherical model and observed results can be significant and has been referred to in literature as *radio irregularity* [60]. This radio irregularity is most

commonly modeled through the use of the log-normal shadowing RF propagation model where the path loss is given by [62]

$$P_r = P_t\left(\frac{\lambda}{4\pi d}\right)^{\alpha} 10^{-\left(\frac{X}{10}\right)} \tag{15}$$

or, equivalently in dB,

$$L_p(d) = L_p(d_0) + 10\alpha \log\left(\frac{d}{d_0}\right) + X \tag{16}$$

where $L_p(d_0)$ is the path loss at the reference distance d_0 and X is a zero mean Gaussian random variable with standard deviation σ. The amount of radio irregularity can then be controlled by σ and the case in which $\sigma = 0$ equates to the spherical model.

It should be noted that this log-normal shadowing model only captures the effect of slow fading (shadowing) on received signal strength. Received signal strength is also impacted by fast fading due to multipath and Doppler effects [59]. When no dominant direct path (line of sight) signal is present, this fast fading is commonly modeled as a Rayleigh random variable of the form [58]

$$f_R(r) = \frac{r}{\sigma^2}\exp\left(-\frac{r^2}{2\sigma^2}\right) \quad \text{for } r \geq 0. \tag{17}$$

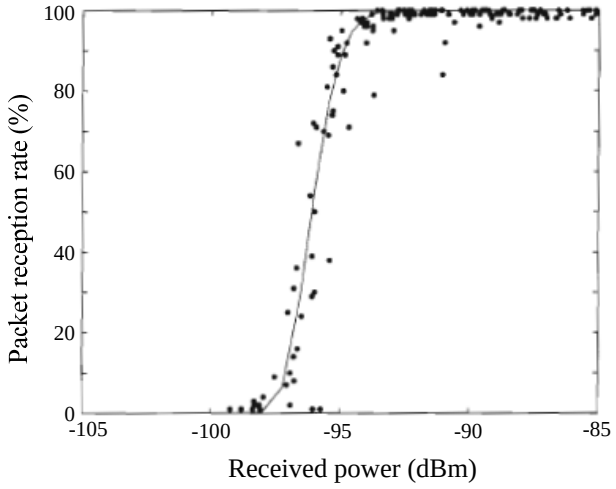
When a direct path signal does exist, a Rician random variable can be used of the form [58]

$$f_R(r) = \frac{r}{\sigma^2}\exp\left(-\frac{r^2 + K^2}{2\sigma^2}\right) I_0\left(\frac{Kr}{\sigma^2}\right) \quad \text{for } r \geq 0, K \geq 0 \tag{18}$$

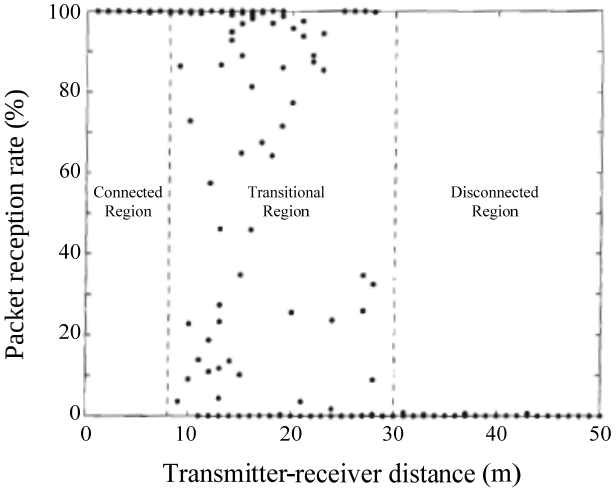
where $I_0(.)$ is the modified Bessel function of order zero and the factor K is defined as the ratio of the power in the dominant path to that in the remaining paths. If $K = 0$, then no dominant signal exists and the channel is Rayleigh. If $K = \infty$, then the channel is additive white Gaussian noise (AWGN) [61].

The bit error rate (BER) and, subsequently, the packet reception rate (PRR) can be derived given a particular MAC framing scheme and physical layer modulation and encoding scheme. PRR is plotted as a function of received power and distance in Figure 7 [62]. Three distinct regions can be seen to exist: (1) the connected

region, (2) the transitional region, and (3) the disconnected region. An important consideration in wireless communications research, this transitional region can be quite large compared to the connected region and the links within this region can be highly variable and asymmetric due to the fluctuating SNR [62].



(a)



(b)

Figure 7. (a) Plot of PRR as a function of received power (indoor). (b) Plot of PRR as a function of distance between transmitter and receiver (From [62]).

b. Interference Model

Two primary interference models exist in literature. The first, commonly referred to as the *protocol model* [63], is an ideal model that assumes packet collisions due to interference at the receiving node cause 100% packet loss. Thus, given an ongoing transmission between nodes *i* and *j* and a common interference range *r*, a transmission from node *k* will result in a collision and subsequent packet loss if the distance between nodes *i* and *k* is less than or equal to *r*. In reality, nodes are capable of correctly decoding packets even in the presence of interference (collisions) provided the received signal to interference plus noise ratio (SINR) is higher than some threshold [62]. Known as the capture effect, this manifests itself in higher observed packet reception rates than predicted by the ideal protocol model. The second common model, referred to as the *physical model* in literature [63], includes this capture effect and successful packet reception requires that the received SINR exceed some threshold $SINR_{th}$ as in [62]

$$\frac{P_s G_{sd}}{\sum_{i \neq s} P_i G_{id} + N_d} > SINR_{th} \tag{19}$$

where P_s is the transmission power at the source node *s*, G_{sd} is the channel gain between the source and destination nodes, and N_d is the noise at the destination noise. The term in the summation represents the interference received at the destination node from all ongoing transmissions (other than that from the source node).

2. Collision Avoidance for Contention-based Approaches

A number of techniques have been proposed in literature to minimize collisions in contention-based medium access schemes. Slotting, one of the earliest techniques, was first proposed in [64] to reduce collisions due to partially overlapping transmissions and slot size was originally based on packet transmission times. With slotting, nodes can only transmit on slot boundaries. TDMA techniques can be seen as a scheduled form of this type of slotted medium access. Slotting was refined in [21] to enhance carrier sensing (which will be discussed next) by reducing the slot size to the maximum propagation delay in the wireless network.

Carrier sensing provides collision avoidance around the sender and was first introduced in [21]. With carrier sensing, the sender senses the medium prior to packet transmission. If the medium is busy, the sender defers. Medium activity can be determined using the signal-to-noise ratio (SNR) at the physical layer and proposed approaches include thresholding [25] and outlier detection [65]. In the former, a single sample is compared to the noise floor and the medium is considered busy if the sample exceeds some threshold. In the latter, multiple samples are taken and the medium is considered free if an outlier is detected that is significantly below the noise floor. Both techniques require the establishment of a good estimate of the noise floor. Carrier sensing can also be accomplished virtually by providing transmission duration information [24] to potentially interfering nodes. With virtual carrier sensing, nodes maintain a counter (called the network allocation vector or NAV in [24]), which is updated based on neighborhood transmission duration information and checked to determine if the medium is busy. Busy tones have also been proposed to provide collision avoidance at the sender [66]. Random backoffs are often used in conjunction with carrier sensing to reschedule deferred transmissions [21] and can be dynamically varied using contention windows, which provide limits on the maximum and minimize size of the backoff [24].

The request-to-send (RTS)/clear-to-send (CTS) mechanism was proposed in [67] to provide collision avoidance around the receiver to combat the hidden node problem. To clear the medium (or *reserve* the floor), the sender transmits a RTS control packet and the receiver responds with a CTS control packet. RTS/CTS packets themselves can suffer from collisions and, to be effective, they must be much smaller than data packets. As in virtual carrier sensing, both packets contain a field that indicates the amount of data to be transmitted in the subsequent data transmission so that neighbors can calculate the duration of the transmission. Upon hearing an RTS, nodes defer to allow reception of CTS. Upon hearing the CTS, nodes defer for the length of the data transmission. RTS/CTS can incur an overhead of 40% - 75% of capacity in WSNs due to the small data packet sizes [23].

3. Distributed Slot Assignment for TDMA Approaches

While it is not specifically addressed in this work, slot assignment is necessary to support packet scheduling in a TDMA-based protocol [68] and can be either fixed or dynamic. The refresh rate of a dynamic scheme is typically based on the rate that traffic demand changes and/or network topology changes (i.e., the faster the load or topology changes, the more often the slot assignment will need to be updated) [19]. Given a slot assignment, schedules can be either "sender only" or both "sender and receiver" [19]. This section defines the slot assignment problem and discusses proposed distributed solutions.

a. The Slot Assignment Problem

For a specific network topology, the slot assignment problem can be defined as finding a transmission slot for each node given the constraint of interference-free transmission [68]. The performance metrics include the maximum number of slots required (and, hence, the frame size), the running time of the slot assignment algorithm, and the number of messages required. This was first formalized in [69] as

$$\begin{aligned} &\text{minimize} && p \\ &\text{subject to} && \left(\frac{1}{p}\right)\sum_{m=1}^{p}\sum_{i\in A_m} x_i^m \geq R, \\ & && x_i^m \alpha_{ik}^m + x_j^m \alpha_{jk}^m \leq 1 \quad \text{for all } m \text{ and for all } k \neq i, j \end{aligned} \tag{20}$$

where p is the number of time slots, R is the minimum desired average transmission rate per slot, and A_m is the set of nodes scheduled for transmission in time slot m,

$$x_i^m = \begin{cases} 1, & \text{if node } i \text{ transmits in time slot } m \\ 0, & \text{otherwise} \end{cases}, \text{ and}$$

$$\alpha_{ij}^m = \begin{cases} 1, & \text{if nodes } i \text{ and } j \text{ are neighbors in time slot } m \\ 0, & \text{otherwise} \end{cases}.$$

In (20), we are minimizing the number of required slots (and, hence, the frame size) given the constraints that the transmission rate is equal to or greater than the desired rate and that the transmissions between nodes do not interfere with each other. This problem equates to the distance-2 coloring problem and has been shown to be NP complete [69].

As can be seen in (20), this interference constraint for a broadcast channel precludes simultaneous transmission between two or more nodes within a single 2-hop "interference" neighborhood [70]. The two-hop interference neighborhood is based on the interference range of the transmitting nodes, which can vary due to path loss variance and transmission power differences [60]. Of particular importance, the interference range is not necessarily equivalent to the communication range (i.e., two nodes may interfere with each other even though they cannot successfully receive transmissions between each other). Hence, the two-hop interference neighborhood is not equivalent to the two-hop communication neighborhood. This is a fundamental challenge to collision avoidance in wireless medium access design.

b. Distributed Slot Assignment and Scheduling

A two-phase solution to the slot assignment problem is proposed in [70]. In the first phase, identified as the "labeling" phase, the order in which nodes will select slots is determined. In the second phase, called the "coloring" phase, nodes select the slots they will transmit in. The latter phase is straightforward; nodes select from a list of currently available slots in a "greedy" fashion (i.e., picking the lowest slot not being used by any of its two-hop neighbors). The labeling can be done in a random order or based on topology (e.g., picking the nodes with the least number of neighbors first). A centralized algorithm is proposed in [70] which uses a master node with full knowledge of the topology to establish the ordering and then the slot assignment is carried out by the master node in a greedy fashion based on this ordering.

The challenge in a distributed solution is to establish the node ordering and determine the set of currently available slots without a central controller. Most approaches in literature accomplish this in successive rounds using some version of the following four-step algorithm, which is based on message exchanges within the two-hop neighborhood.

(1) *With some probability, nodes attempt to claim a slot*. This is typically accomplished through a broadcast message to all of their one-hop neighbors.

(2) *Nodes determine if they are successful in claiming a slot.* Nodes compile all of the messages they receive from step (1) and combine them in a single message, which is then rebroadcast to all of their one-hop neighbors. This has the effect of informing all nodes of the "claims" throughout their two-hop neighborhood. If there are no conflicting claims, then a node is successful in claiming a slot.

(3) *If they are successful, they inform their one-hop neighbors*. Again this is done through a broadcast message. Upon receiving this "successful" message, nodes remove the appropriate slot from their list of available slots.

(4) *The one-hop neighbors inform the two-hop neighbors*. Nodes compile all of the messages they receive from step (3) and combine them in a single message, which is then rebroadcast to all of their neighbors. This has the effect of informing all nodes of the "successful claims" throughout their two-hop neighborhood. Upon receiving this "successful" message, nodes remove the appropriate slot from their list of available slots.

This four-step algorithm is repeated until all nodes have been assigned a slot. It should be noted that this four-step algorithm bears some resemblance to the RTS/CTS mechanism. In addition to determining the slot number for each node, the frame size (i.e., total number of slots) must also be disseminated and, in a wireless channel, the algorithm must accommodate packet/message losses as well as node failures. Most published schemes either assume a fixed frame size or make use of a central controller to determine and disseminate frame size changes. Effective and efficient dynamic frame size in a distributed network remains an open research question. Performance measures of a distributed solution include maximum number of slots (and, hence, frame size), time to convergence, probability that all nodes will be assigned a slot, and probability that nodes will be assigned "conflicting" time slots. In the following, we review a few of the more prominent and representative proposals in literature.

The Five-Phase Reservation Protocol (FPRP) [71] is a TDMA-based MAC protocol. Time is divided into reservation frames and information frames where the former is used to reserve data time slots in the latter. A reservation slot is provided for each data slot. Nodes with traffic to transmit contend with some probability for each slot in its corresponding reservation slot based on the message exchange above. If a node is successful in a reservation slot, it claims the corresponding data slot. Claims are deemed successful if a node does not receive a collision report from any of its one-hop neighbors in step (2). Successful reservations are promulgated through the two-hop neighborhood as in steps (3) and (4). A fifth step is added to eliminate any potential conflicts (termed "deadlocks") due to message losses in step (2) and speed up the convergence time by promulgating successful reservations to the three-hop neighbors. The maximum number of slots and the number of rounds required to successfully complete the reservation process within each reservation slot (and, accordingly, the convergence time) are both determined heuristically and are fixed during runtime.

DRAND [68] is a distributed version of the RAND slot assignment scheme [70] and serves as the scheduling mechanism for the hybrid Z-MAC protocol [23]. DRAND enters step (1) above with probability $1/k$ where k is the node's estimate of the number of one- and two-hop neighbors who have not been assigned a slot yet. Nodes broadcast a "request" message to all of their neighbors. This message is a request to claim a slot (vice actually identifying which slot it is trying to claim). In step (2), neighbors respond with grant messages if they have not received any other request messages. If a node receives grant messages from all of its one-hop neighbors, it assumes it's request is successful and selects the lowest unused slot (in its two-hop neighborhood) based on information provided in the grant messages. This assumes that a node has knowledge of all its one-hop neighborhood. Finally, release messages are exchanged in steps (3) and (4) to inform the two-hop neighborhood of the slot assignment. Messages losses are handled using a retransmission mechanism and node failures are addressed through the use of a timeout mechanism. The maximum number of slots is bounded by $\delta+1$ where δ is the maximum size of the two-hop neighborhood. Experimental results indicate that the number of slots can be well below this maximum, but no mechanism is provided to

disseminate the actual number of slots utilized at runtime. Convergence time and message complexity are both $O(\delta)$ for DRAND. Experimentally, DRAND was shown to outperform FPRP (using both 10 and 50 reservation cycles per reservation slot) in terms of the maximum number of slots utilized and message complexity. Convergence time performance is dependent on the number of reservation cycles chosen for FPRP.

The D2-coloring algorithm of [72] also uses the four-step algorithm above with the modification that the initial "claim" (called a TRIAL message) is broadcast in the appropriate slot in step (1). This allows multiple nodes within a single two-hop neighborhood to claim different slots in the same round. Compiled reports (called TRIAL-REPORT messages) are transmitted in step (2) and if a node hears its TRIAL in the TRIAL-REPORT messages of all of its one-hop neighbors, then it was successful (no one else tried to claim the same slot in the two-hop neighborhood and, hence, there were no collisions) and can claim the slot. The success is reported in subsequent SUCCESS and SUCCESS-REPORT messages as in steps (3) and (4). The frame size (i.e., the number of slots) is fixed prior to runtime and the algorithm handles message losses by transmitting the TRIAL-REPORT and SUCCESS-REPORT messages multiple times by randomly selecting a slot in each of multiple frames. Similar to the reservation cycles in FPRP, the number of frames in steps (2) and (4) impact the convergence time for this algorithm. The number of slots, total number of rounds, and the number of frames in steps (2) and (4) are all chosen heuristically and fixed. Nodes must know their one-hop neighborhood and must be able to estimate the total number of nodes in the network, *n*, as well as the maximum degree, Δ. The convergence time for this D2-coloring algorithm is $O(\Delta \log^2 n)$ and the maximum number of messages is $O(n \log^2 n)$.

The traffic-adaptive medium access protocol (TRAMA) [19] is a TDMA-based MAC protocol that takes a different approach to scheduling. Node scheduling is resolved using a local contention resolution algorithm based on the neighborhood-aware contention resolution (NCR) algorithm [73]. This algorithm is run locally at each node and "winners" (nodes that can transmit in the given slot) are uniquely determined within each two-hop neighborhood from priorities set by a hash function of the node ID and the time slot number. To properly execute this distributed election algorithm, all nodes must

have a unique ID, this ID must be known throughout the two-hop neighborhood, and the nodes must be synchronized (time slots must be aligned at all nodes). Time is divided into contention and contention-free periods and the data is transmitted in time slots within the contention-free period. The protocol is comprised of three components. The Neighbor Protocol (NP) uses contention-based medium access and obtains two-hop neighbor knowledge at all nodes. In the contention-free period, the Schedule Exchange Protocol (SEP) is used to promulgate sender-receiver schedules for all nodes across their two-hop neighborhoods. Given the information from NP and SEP, nodes locally determine both transmitters and receivers for each slot through the Adaptive Election Algorithm (AEA), which is based on the NCR algorithm. Nodes are allowed to sleep if they are neither a designated transmitter nor receiver in a given slot. The total number of slots is fixed, although nodes only contend for slots if they have traffic to transmit. AEA ensures that the locally calculated winners are consistent across two-hop neighborhoods. The flow-aware medium access (FLAMA) protocol [74] is a follow-on to TRAMA that uses the tree structure of data gathering applications to develop flow-based weights that are then included in the distributed election algorithm. Additionally, the formation of the tree allows FLAMA to dispense with the schedule exchange phase. FLAMA is not appropriate for peer-to-peer, non-tree-based communication flows.

4. Time Synchronization

Time synchronization is required to support application level event synchronization, sleep cycle coordination, and scheduled medium access schemes [75]. Again, although it is not the focus of this work, time synchronization will be necessary to support the proposed TDMA-based medium access solution. This section provides an overview of the fundamentals of time synchronization followed by a survey of common techniques and associated protocols.

a. Time Synchronization Fundamentals

A digital clock is typically comprised of a counter that is triggered by an oscillator [76] and can be modeled as [77]

$$C_i(t) = \kappa \int_{t_0}^{t} \omega(\tau) d\tau + C_i(t_0) \tag{21}$$

where $C_i(t)$ is the "local" time at node *i*, *t* is the "real" time, ω is the angular frequency of the oscillator, and κ is a constant associated with the oscillator. If we assume the oscillator has a constant angular frequency and set t_0 to zero, (21) reduces to [78]

$$C_i(t) = a_i t + b_i \tag{22}$$

where a_i and b_i are the clock drift and clock offset relative to real time at node *i*, respectively. The drift can be seen to be the rate at which the local clock time changes with respect to the real time, as in

$$a_i = \frac{dC_i(t)}{dt}. \tag{23}$$

The objective of time synchronization then is to set $C_i(t) = C_j(t)$ for some set of nodes *i* and *j* where $i \neq j$.

We can use (22) to compare two clocks to arrive at

$$C_i(t) = a_i \left[\frac{C_j(t) - b_j}{a_j} \right] + b_i. \tag{24}$$

Defining $\frac{a_i}{a_j}$ as the relative drift a_{ij} between the clocks and $b_i - \frac{a_i}{a_j} b_j$ as the relative offset b_{ij}, we have

$$C_i(t) = a_{ij} C_j(t) + b_{ij}. \tag{25}$$

Synchronization then equates to a relative drift of one and a relative offset of zero. Some published work uses the term "skew" vice "drift," typically to capture the difference vice the ratio of the clock rates [78]. Examining (25), we can identify the ways that two clocks can be "out-of-sync:"

- *Relative offset*. Two clocks will be out-of-sync if their initial values are not the same (relative offset not equal to zero).
- *Relative drift.* Even if the relative offset is corrected, two clocks will become unsynchronized if their oscillator frequencies are not the same (relative drift not equal to one or, equivalently, their skew is not equal to zero). Sensor nodes typically use inexpensive crystal oscillators whose frequencies can

differ by as much as 100 parts per million (ppm), which is equivalent to a relative drift of 100 μs per second [76]. MICA2 motes are capable of frequency differences as large as 40 μs per second [79].

- *Stability (or drift variation).* As discussed earlier, (22), and consequently (25), assume that the oscillator frequency is constant. In reality, oscillator frequency can vary over time. Short-term frequency instability is primarily the result of environment conditions such as changes in temperature or supply voltage while oscillator aging is a common cause of long-term instability [80].

Time synchronization solutions can be classified by their scope and fidelity. Global solutions synchronize all nodes within a network to a common reference time while local solutions synchronize a subset of nodes. The fidelity of synchronization models can be divided into three classes: ordering, relative, and "always on" [81]. The simplest approach is to maintain relative order between successive events at different nodes. This ordering approach can be viewed as time synchronization only in the broadest sense and will not be explored here. Relative synchronization strategies, the most common approach in WSNs, allow local clocks to run unsynchronized, but maintain enough information to convert the local time of one node to the local time of another node of interest. "Post-facto synchronization" [82] is an example of relative synchronization where events are recorded in local time and nodes are synchronized immediately following the event to relate the local time readings. The final, most complex, approach is the "always on" model where all nodes maintain a local clock that is always synchronized to a reference time.

Most time synchronization methods involve the exchange of timestamp information between nodes [78]. Nondeterministic delay in this message exchange poses a fundamental challenge to time synchronization solutions. As shown in Figure 8, the uncertainty in delay can be divided into the following components [79]:

- *Send time*. This is the time required to construct the message and transfer it to the MAC level. Caused by the kernel processing, context switches, and system

calls in the operating system, the send time is non-deterministic and depends on processor load. Typical values can be as high as 100 ms.

- *Access time*. This is the time spent waiting for the channel to become available for packet transmission. It is non-deterministic and depends on the medium access scheme and the contention level of the network. Typical values range from 10 – 500 ms.
- *Transmission time*. This is the time required to transmit the message (bit by bit) and it overlaps with the subsequent propagation and reception times. Generally deterministic, it is dependent on message size and radio speed. It does contain some level of non-determinism due to small variations in interrupt handling times. Typical values are 10 – 20 ms.
- *Propagation time*. This is the time it takes for a packet to travel across the wireless link from the sender to the receiver. It is deterministic and depends on the distance between nodes. Typical values are less than 1 μs for distances less than 300 m.
- *Reception time*. The time it takes to receive the message and forward it to the MAC layer. Generally deterministic, it is dependent on message size and radio speed. It does contain some level of non-determinism due to small variations in interrupt handling times. Typical values are 10 – 20 ms.
- *Receive time*. Similar to the send time, this is the time required to reconstruct, forward and decode the message. It is non-deterministic, depends on processor load, and typical values can be as high as 100 ms.

As we shall see in the next section, time synchronization techniques can be compared by the approaches employed to overcome these message delay uncertainties.

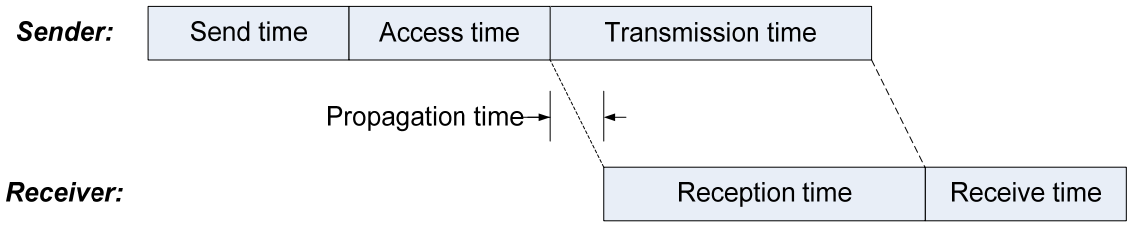


Figure 8. Message delays (After [79]).

b. Time Synchronization Techniques and Protocols

The simplest message exchange scheme is one-way transmission from the time synchronization server to the client. The advantage of this approach is that a single message can be broadcast to synchronize multiple clients, so the communication overhead is low [76]. The main disadvantage is that the client cannot estimate the non-deterministic portion of the message delay. The Flooding Time Synchronization Protocol (FTSP) [79] is an "always on" approach that utilizes this one way message exchange and mitigates much of the message delivery uncertainty by timestamping at the MAC layer just prior to transmission. In addition, it uses multiple timestamps per message to remove uncertainty in the transmission and reception times. This MAC layer timestamping implicitly assumes access to the MAC layer. FTSP is a global time synchronization protocol that uses a root node to flood the network with time synchronization information. In addition to offset synchronization, it achieves drift estimation using linear regression on the eight most recent synchronization sample points. Linear regression assumes a linear relationship between samples, which implies that the drift is constant. Experimental results demonstrated an average single hop synchronization error of 1.48 μs and an average multi-hop error of 0.5 μs per hop for FTSP.

The Reference Broadcast Synchronization (RBS) scheme [80] introduces a receiver-receiver technique that utilizes a beacon to synchronize multiple receivers on demand. Receivers exchange local reception times to determine relative offsets. By comparing receiver reception times, RBS effectively removes the uncertainty in the sender's send time, access time, and transmission time from the critical path. Treating differences in reference pulse propagation times as negligible, this leaves only the

uncertainty in the receiver's reception and receive times. As in FTSP, RBS uses linear regression across multiple samples to estimate rate differences. Although single hop by definition, a multi-hop solution is also provided in [80] that utilizes common nodes between adjacent single-hop broadcast regions to establish global timescales. The authors report a timestamping accuracy of 11 μs on MICA motes, which was reduced to 7.4 μs by accounting for clock drift using linear regression over a 60 s interval. Multi-hop synchronization error was of $O\left(\sqrt{n}\right)$ for n hops.

The Timing-sync Protocol for Sensor Networks (TPSN) [81] is an "always on" solution that makes use of a two-way sender-receiver message exchange approach that relies on a hierarchical structure and performs pairwise synchronization along the edges. The two-way message exchange enables message delay estimation and it uses MAC layer timestamping to remove the highly variable access time delay. Offset calculation is based on timestamp information exchanged in the request and acknowledgement messages. TPSN does not provide estimation of drift and has the added overhead of the acknowledgement message when compared to the one-way exchange of FTSP. The authors prove a 2x performance improvement over RBS and argue that previously published RBS results are an artifact of the operating system used. On their MICA mote implementations, they report average synchronization errors for a single-hop network of 16.9 μs for TPSN and 29.1 μs for RBS.

The Network Time Protocol (NTP) [83] is the widely accepted Internet time synchronization protocol. In NTP, nodes are established in a hierarchy and client (leaf) nodes synchronize their clocks via a round-trip message exchange with a preconfigured server. Nodes maintain synchronization by periodically updating their system clocks based on the information provided in the frequent synchronization exchanges. NTP uses a phase lock loop to estimate and correct for variable drift. Accuracy of NTP is on the order of milliseconds [79]. It is not well suited to WSN applications because it requires that the nodes establish a preconfigured hierarchy, remain "awake" to execute regular clock updates, constantly listen for synchronization requests

from clients, and it attempts to accurately estimate delays across multiple hops, which can be highly variable in a wireless network [76].

The Global Positioning System (GPS) is a satellite-based system that can provide an external timing reference with a synchronization accuracy of 200 ns [84]. GPS receivers, however, are still relatively expensive, energy inefficient, and require direct line of sight to several satellites [76]. Accordingly, GPS cannot be used in heavy foliage or inside buildings and may not be appropriate for small, low cost sensor mote solutions.

5. Power Management at the MAC Layer

Power consumption is a significant challenge in both MANETs and WSNs due to the limited battery power available [85],[30]. This is particularly exacerbated in WSNs where it may be impractical to change out or recharge the sensor node batteries [30]. As discussed previously, communication consumes more energy than processing and computation and there is an opportunity to trade more on-board or in-network processing for less internode communication. As an example, for a MICA2 mote the energy cost is 720 nJ/bit to transmit versus 4nJ/operation [31]. The authors of [22] and [30] identify five major sources of energy waste in wireless communications. Packets must be discarded and retransmitted when they experience *collisions*. The retransmissions result in an increase in both power consumption and latency. There has been some research into exploiting capture effect to recover a packet despite a collision [86],[87]. *Overhearing* occurs when nodes receive packets for which they are not the intended receiver. *Control packet overhead* stems from the use of dedicated control packets to coordinate transmissions. *Idle listening* occurs when nodes listen for packets while the channel is idle. Finally, *overemitting* occurs when a message is transmitted to a destination that is not ready to receive it.

Power management can be divided into two categories. Power save techniques attempt to minimize the power loss due to the communication issue described in the previous paragraphs. These approaches typically involve powering down the transceiver in a “sleep state.” Power control techniques, in contrast, attempt to minimize power consumption by directly addressing the transmit power used in each transmission. This section provides a brief overview and discusses some of the significant proposals in

literature for both the power save and the power control approaches. To provide a framework, it begins with a survey of existing energy consumption models.

a. Energy Consumption Model

The first and most common energy model seen in WSN research was introduced in [88] for free space path loss ($\alpha = 2$) and further developed in [89] to also include multipath ($\alpha = 4$). In this energy consumption model, based on the radio model of Figure 9, the total transmission energy is

$$E_{tx} = \begin{cases} k\left(E_{elec} + \varepsilon_{fs} d^2\right) & d < d_0 \\ k\left(E_{elec} + \varepsilon_{mp} d^4\right) & d \geq d_0 \end{cases} \tag{26}$$

and the total reception energy is

$$E_{rx} = kE_{elec} \tag{27}$$

where k is the number of bits and E_{elec} is the energy consumption due to the transmitter or receiver electronic circuitry (assumed in [89] to be equivalent for both the transmitter and receiving circuits). The threshold distance, d_0, is the distance at which the channel model switches between the free space model with the amplifier factor ε_{fs} to the multipath model with the amplifier factor ε_{mp}. It should be noted that the per bit energy required for transmission is of the form $a + bd^{\alpha}$ where a is a constant, distance independent term. The authors of [90] point out that for short range radios, the distance independent term a is typically much larger than the distance dependent term bd^{α}.

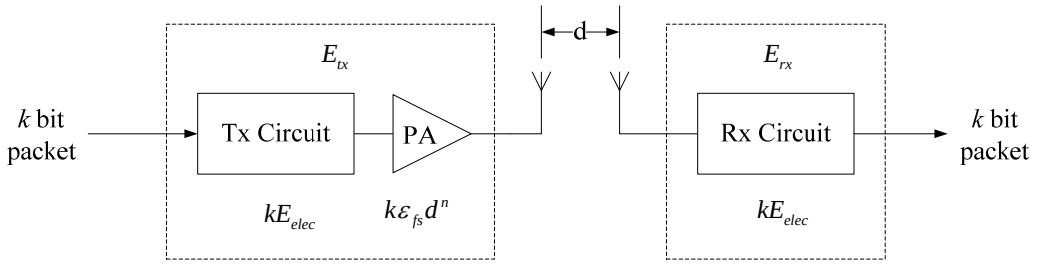


Figure 9. Radio model for energy consumption model of [88] and [89] (From [89]).

In [91], the authors utilized the more detailed radio model of Figure 10 to include energy consumption due to startup and further develop the transmission and

reception energy costs. The energy required to initially power up the transceiver is due to the time t_{start} it takes the frequency synthesizer and the VCO to lock onto the carrier signal and is modeled as

$$E_{start} = \left(P_{fs} + P_{VCO} \right) t_{start} \tag{28}$$

where P_{fs} and P_{VCO} are the power consumption associated with the frequency synthesizer and the VCO, respectively. The receive energy is modeled as

$$E_{rx} = \left(P_{fs} + P_{VCO} + \mathrm{P}_{rx} \right) t_{rx} \tag{29}$$

where t_{rx} is the reception time and P_{rx}, which is assumed to be constant, includes the power consumption of the low noise amplifier (LNA), mixer, intermediate frequency amplifier, and the demodulator. Finally, the transmission energy consumption for a desired $\frac{E_b}{N_0}$ and link margin L_m at the receiver is modeled as

$$E_{tx} = \left(P_{fs} + P_{VCO} + \frac{1}{\eta} \left(\frac{\left(4\pi\right)^2 N_0 F L_m}{G_T G_R \lambda^2} \right) \left(\frac{E_b}{N_0} \right) R d^{\alpha} \right) t_{tx} \tag{30}$$

where η is the efficiency of the amplifier (defined as the ratio of the output power to the input power), F is the noise factor and G_T and G_R are the transmitting and receiving antenna gains, respectively. Again, this per bit transmission energy is of the form $a + bd^{\alpha}$ with

$$a = P_{fs} + P_{VCO} \tag{31}$$

and

$$b = \frac{1}{\eta} \left(\frac{\left(4\pi\right)^2 N_0 F L_m}{G_T G_R \lambda^2} \right) \left(\frac{E_b}{N_0} \right) R . \tag{32}$$

In all cases, [91] assumes that the energy consumption of the digital signal processing unit (encoding/decoding) as well as the A/D and D/A converters is negligible.

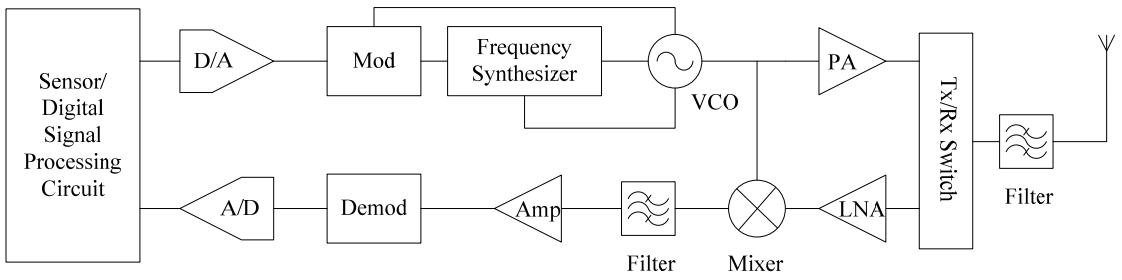


Figure 10. Radio model for energy consumption model of [91] (From [91]).

Utilizing the functional radio model of Figure 11, the authors of [92] proposed a multi-hop energy consumption model that also included the energy consumption of the baseband digital signal processor. For path of *n* hops, the total power consumption is

$$P_{total} = (n-1)P_{rx} + nP_{tx} \tag{33}$$

where

$$\begin{aligned} P_{rx} &= P_{RB} + P_{RRF} + P_L \\ P_{tx} &= P_{TB} + P_{TRF} + \left(\frac{P_{rx}G_TG_R}{\eta}\right)\frac{1}{n}\sum_{i=1}^{n} d_i^{\alpha} \end{aligned} \tag{34}$$

and P_{RB}, P_{RRF}, P_{TB} and P_{TRF} are defined as in Figure 11. The authors of [92] point out that (1) the power consumption of the low noise amplifier P_L in the receiver can be modeled as constant provided it is setup for a minimum received power and (2) the transmitter amplifier efficiency typically increases with increasing output power.

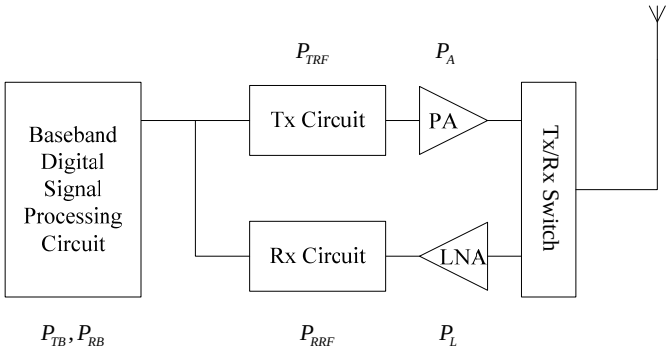


Figure 11. Radio model for energy consumption model of [92] (From [92]). P_{TB} and P_{RB} are the power consumption in the baseband for transmitting and receiving, respectively.

The authors of [93] express their energy consumption terms using current and voltage values, which can be found in the specification sheets for most sensor nodes. In addition to the transmitting and receiving energy, they include energy consumption expressions for both the idle and active microcontroller unit (MCU) states, the transceiver idle (listening) state, the transceiver sleeping state and the switching energy required to transition between states. In all cases, the energy consumption is the published (or measured) current draw in the state multiplied by the supply voltage and the time spent in the state. Measured current consumption for Telos, Mica2 and MicaZ motes [94] are shown in Table 2. Worth noting, it can be clearly seen that the receive power (at 0 dBm) is roughly equivalent to (and in some cases, greater than) the transmit energy for the newer generation motes. This has a significant impact on the design of low-power MAC protocols.

Operation	**Telos**	**Mica2**	**MicaZ**
Minimum voltage	1.8V	2.7V	2.7V
Mote standby	5.1 μA	19.0 μA	27.0 μA
MCU idle	54.5 μA	3.2 mA	3.2 mA
MCU active	1.8 mA	8.0 mA	8.0 mA
MCU + radio RX	21.8 mA	15.1 mA	23.3 mA
MCU + radio TX (0 dBm)	19.5 mA	25.4 mA	21.0 mA
MCU + flash memory read	4.1 mA	9.4 mA	9.4 mA
MCU + flash memory write	15.1 mA	21.6 mA	21.6 mA
MCU wakeup time	6 μs	180 μs	180 μs
Radio wakeup time	580 μs	1800 μs	860 μs

Table 2. Measured current consumption for the Berkeley family of motes (After [94]).

b. Power Save Modes

IEEE 802.11 [24] includes a mechanism to power down a node into a reduced power “sleep state” called the power save (PS) mode. Specifically, the transceiver is powered down in the PS mode and a node can neither transmit nor receive. This PS mode is implemented as follows. In the infrastructure mode, the access point (AP) is always on and coordinates traffic for the mobile host in the PS mode. The mobile

host informs the AP when it is powering down and the AP then buffers packets for the host in PS mode. The mobile host periodically wakes up to check if the AP has buffered packets for it. In the ad hoc mode, each mobile host in the PS mode wakes up periodically during designated ad hoc traffic indication map (ATIM) windows. Within these ATIM windows, nodes transmit ATIM messages (using the DCF mechanism), which indicate intended receivers for buffered packets. If a host in PS mode is not on the list of intended receivers, it will power down until the next ATIM window. IEEE 802.11 is designed for single-hop (or fully connected networks where all nodes can "hear" each other) and synchronization, neighbor discovery and network partitioning present problems when this scheme is applied to multi-hop ad hoc wireless networks [95]. The authors of [95] present several proposals to address these issues, but the control overhead and latency of their proposed solutions can be large because they do not coordinate the sleep periods of the nodes [22]. The Power Aware Multi-access protocol with Signaling (PAMAS) [96], which proposes a second signaling channel, also takes advantage of the RTS/CTS exchange to power nodes down if they are not the intended receiver of the upcoming transmission. The contention resolution mechanism of all of these protocols reduces the energy waste due to collisions and the sleep mechanism limits the overhearing cost.

While the protocols discussed above reduce the energy consumption due to overhearing, they do not address the problem of idle listening. In an ideal solution to this problem, a node will only wakeup when it is the destination for the upcoming packet transmission. The basic idea, then, is that nodes must have very low duty cycles and the state of the art is on the order of 0.1% [97]. In the words of sensor network pioneer David Culler from UC Berkeley during his keynote speech at SECON 2008, the key is to "do nothing well" [97]. The challenge of low duty cycle operation has been approached through both asynchronous and synchronous techniques.

There are two common asynchronous approaches to sleep implementation. The first is a hardware solution that makes use of a secondary, low-power "wake-up radio" [98],[99] while the second is an algorithmic solution that is referred to as either *preamble sampling* or *low power listening* in literature [38],[100]. The requirement for a

second radio is an implementation issue for WSNs, so we turn our attention instead to the latter approach. In preamble sampling, a node will periodically wakeup and listen to the channel to see if it has traffic pending. Thus, a node with a packet to transmit need only transmit a beacon for the duration of the sampling cycle to wake-up the destination node. Upon waking up and hearing the beacon, a node with then remain awake to receive the subsequent transmission. The beacon can simply be a physical layer RF pulse, which is easy to implement, but this has the undesirable side effect that all nodes in the reception range will remain awake and wait for the subsequent transmission to determine whether they are the intended destination. At higher loads, this can result in a substantially higher duty cycle. Alternately, the beacon can be a MAC layer mechanism that includes destination information, but this requires a more complex implementation. WiseMAC [101] reduces the requirement for the relatively long preamble transmissions by allowing neighboring nodes to exchange preamble sampling times. The preamble duration is then a function of the accuracy of the synchronization between the two nodes (though bounded by the preamble sampling period). Berkeley Media Access Control (B-MAC) [65], a commonly used reconfigurable MAC protocol that has been implemented on the Berkeley family of motes, includes preamble sampling in the suite of functions it provides.

By synchronizing sleep schedules, a family of protocols [22],[102],[103], have been proposed that further reduce the energy consumption due to idle listening. S-MAC [22] addresses the idle listening problem and attempts to improve control overhead and latency by coordinating the sleep periods of neighboring nodes. This is accomplished through the use of sleep schedules, which are broadcast among neighbors. Neighbors then form virtual clusters by aligning their sleep schedules. The result is a set of coordinated, fixed length, sleep (or duty) cycles that are comprised of alternating periods of listening/transmitting and sleeping. Timeout-MAC (T-MAC) [102] improves upon the energy-efficiency of this scheme by allowing the sleep cycle to be adaptive through the use of an inactivity time-out mechanism. D-MAC [103] takes advantage of the data-gathering tree structure in many WSNs to coordinate the sleep schedules and reduce the latency introduced by the sleep cycles.

Scheduled access using contention-free, TDMA approaches eliminates the energy cost due to collisions and also allows nodes to sleep when they are neither transmitting nor receiving in a given time slot. To realize these latter energy savings, nodes must be able to determine if they are the intended receiver in the transmission scheduled in the given slot. This can be accomplished through the use of sender and receiver scheduling as in [19],[104] or preamble sampling as in [23] where nodes wakeup at the beginning of each slot and check to see if they are the intended receiver.

An important consideration in the utilization of sleep modes is that the startup cost associated with the transition from the sleep state to the active state [85] could offset the potential savings achieved in the sleep mode. Thus, it is important to consider not only the total sleep time but also the duration of an average sleep period, which provides a metric to reflect the impact of the number (or frequency) of state transitions [19].

c. Power Control Techniques

The transmit power level impacts the received signal strength at the destination node, the range of the transmission, and the magnitude of the resulting interference at neighboring nodes [105]. It can be used to control topology by varying transmission power node by node to affect connectivity or it can also be used on a packet-by-packet basis to improve energy consumption and/or throughput [106]. In this section, we will focus on the latter and present a number of energy-efficient and throughput-oriented protocols that have been proposed in literature. The section begins with a discussion of the shortfalls of the fixed power IEEE 802.11 medium access scheme.

The fixed transmission power scheme of IEEE 802.11 [24] suffers from reduced throughput, increased delay and increased energy consumption due to the excessively large reservation area associated with the maximum power transmission of the data packets and the protocol definition of a collision [107]. The first point can be clearly seen in Figure 12 where, at maximum power, the data transmission between nodes A and B prevents data transmission between nodes C and D. Alternately, if nodes A and B use the minimum power needed for effective communication, both transmissions can occur concurrently. Related to this point, the second shortfall in the IEEE 802.11

approach assumes that interference implies a collision and subsequent packet loss. As discussed earlier, this is a simplified interference model, which results in a conservative approach, which leads to decreased channel utilization. Using the more realistic physical model of (19), it can be seen that a packet can be captured, or correctly received, provided that the SINR is above some threshold. Hence, even though the transmission from A to B can be sensed at C and D in Figure 12, it may not result in subsequent packet loss. The goal of transmission power control schemes then is to reduce the transmission power to allow concurrent transmissions provided that the packets can be correctly received given the current interference levels at the destination nodes.

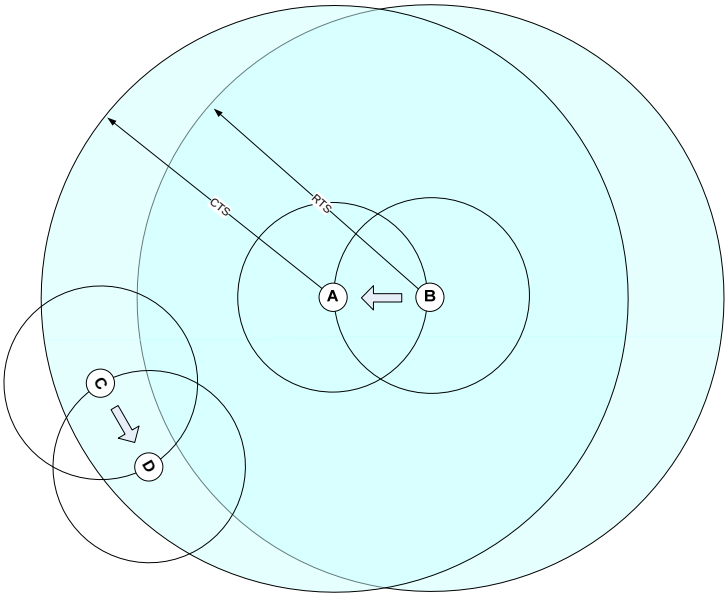


Figure 12. The need for transmission power control to allow concurrent transmissions (From [108]).

Several energy-oriented power control protocols [67],[109] have proposed enhancements to IEEE 802.11 to adaptively reduce the transmission power using the RTS/CTS exchange. RTS/CTS packets are transmitted at maximum power to prevent collisions that may occur when node transmission power is not uniform and subsequent data packets are then transmitted at the minimum power required for successful communication between the transmitting and receiving nodes. The RTS/CTS exchange can be used to determine the minimum required transmission power at the sender by

including the power level for the RTS transmission, $P_t^{(RTS)}$, in the RTS packet [110]. This allows the destination node to calculate the channel gain for the sender to the destination, G_{sd}, as $G_{sd} = P_t^{(RTS)} / P_r^{(RTS)}$ where $P_r^{(RTS)}$ is the received power level of the RTS packet at the destination. Given the minimum signal-to-noise ratio required for reliable communication, $SINR_{th}$, the destination can then calculate the required data packet transmission power at the sender as

$$P_{\min}^{(data)} = \frac{SINR_{th} P_{noise}^{(d)}}{G_{sd}} = \frac{SINR_{th} P_{noise}^{(d)} P_t^{(RTS)}}{P_r^{(RTS)}} \tag{35}$$

where $P_{noise}^{(d)}$ is the noise power measured at the destination. This minimum required transmission power is then forwarded to the sender in the subsequent CTS packet. An underlying assumption here is that the channel gain is constant for the duration of the RTS/CTS/Data/ACK exchange. Alternatively, if we also assume a symmetric channel with the same gain in both directions (i.e., $G_{ds} = G_{sd}$), $P_{\min}^{(data)}$ can be calculated at the sender as [109]

$$P_{\min}^{(data)} = \frac{SINR_{th} P_{noise}^{(d)} P_t^{(CTS)}}{P_r^{(CTS)}} \tag{36}$$

if $P_{noise}^{(d)}$ and the power level for the CTS transmission, $P_t^{(CTS)}$, are included in the CTS packet. Although energy consumption in theses schemes is reduced due to the lower data transmission power, the throughput is at best the same as IEEE 802.11 because the full power RTS/CTS mechanism still silences all neighboring nodes out to the maximum transmission range [106].

Interference-aware protocols [111],[110],[112],[106] attempt to further improve throughput by allowing concurrent transmissions provided that they do not disrupt ongoing transmissions. By advertising maximum allowable interference information, the RTS/CTS exchange bounds the transmission power for neighboring nodes rather than silencing them. For this approach to be effective, an interference margin must be built into the transmission power calculation for data (and ACK) packets (i.e., these packets must be transmitted at a power higher than the minimum power required for reliable transmission) [107]. Thus,

$$P_{\min}^{(data)} = \frac{SINR_{th}\left(P_{noise}^{(d)} + P_{\text{int}}^{(d)}\right)}{G_{sd}} \tag{37}$$

where $P_{\text{int}}^{(d)}$ is the maximum allowable noise from interfering transmissions. Assuming the gain in the channel between the destination and the potential interfering neighbor node is the same in both directions (i.e., $G_{id} = G_{di}$), the potential interfering neighbor node can then use $P_{\text{int}}^{(d)}$ (which can be published in the CTS) to bound its transmission power for any overlapping transmission as in [106]

$$P_{t,bound} = \frac{P_{\text{int}}^{(d)}}{G_{id}}. \tag{38}$$

The schemes proposed in [110], [111], and [112] all use a separate control channel to advertise maximum allowable interference information. In Power Controlled Medium Access (PCMA) [110] and Intelligent Medium Access (IMA) for MANETs [111], busy tones are transmitted on the control channel to bound subsequent transmissions while the interference information in the Power Controlled Dual Channel (PCDC) medium access protocol [112] is explicitly encoded in the RTS/CTS packets, which are then transmitted on the control channel. More recently, the authors of the power control MAC (POWMAC) [106] offer a single channel interference aware solution in which the maximum allowable interference information is provided in the RTS/CTS packets and multiple RTS/CTS exchanges for concurrent transmissions are grouped together in a single access window to further improve throughput. Compared to IEEE 802.11, POWMAC is shown to increase throughput by 30-40% for random grid topologies and by over 50% for clustered topologies. Energy consumption is comparable to IEEE 802.11 for the random grid, but POWMAC shows significant improvement in the clustered topology because most communications occur within the cluster, which requires substantially less then the maximum transmission power.

In closing this section, we note that the authors of [105] present the following fundamental design principles that can help guide the development of energy efficient power control schemes.

- Reducing transmit power level increases network capacity.

- Reducing transmit power level reduces MAC layer contention.
- When a uniform power level is used, there exists a critical transmission range below which transmissions are suboptimal in terms of energy consumption. This range is given by

$$R = \sqrt{\frac{P_{rx} + P_{tx}}{c(\alpha - 1)}}. \tag{39}$$

- At high network loads, lower transmit power levels provide lower end-to-end delay while higher power levels provide lower delay when the load is low.

6. Cross-layer Design Overview

Layered and cross-layer approaches can be broken down into three groups based on the parameters to be optimized and the information used in the optimization. In a strictly layered design, optimization is achieved within a single layer using only the information available within that layer. In a loosely coupled design [113], optimization is carried out within a single layer but information is provided across layers to improve the solution. In a tightly coupled design [113], parameters in different layers are jointly optimized using cross-layer information sharing. One would expect that a tightly coupled design would result in a "better" solution [113], but it comes at the cost of increased overhead in cross-layer communication requirements and the increased computational burden of joint optimization of multiple variables. Cross-layer design approaches can also be classified as top-down or bottom-up based on the order of the optimization [114].

The authors of [115] caution against "unbridled" cross-layer design and discuss the "fundamental tension between performance and architecture," which can be viewed, in some respects, as a trade-off between short-term gains and long-term gains, respectively. Layered architecture provides the modularization necessary to decompose the overall problem and allow parallel development and implementation. This facilitates rapid proliferation and longevity. Examples of successful layered architectures include the von Neumann architecture, which decouples software and hardware, the current Internet architecture derived from the OSI model, and Shannon's source separation theorem, which decouples source coding and channel coding. Cross-layer solutions can lead to unintended consequences and the designers of cross-layer solutions must be aware

of dependencies and interactions with other protocols at all layers. To address this, the authors recommend the use of dependency graphs and timescale separation to ensure stability of cross-layer proposals. As an example, [115] demonstrates that the use of a rate-adaptive opportunistic MAC scheme (which increases the data rate when the channel is good) can result in a performance reduction when combined with a minimum-hop routing scheme because the routing scheme will favor longer hops, which have lower SNR and therefore result in lower data rates.

Given an existing layered architecture, the first step in approaching a potential cross-layer solution is to identify the information available at each layer and the parameters that can be optimized or "tuned." Table 0 presents a listing of some common cross-layer opportunities in the context of the existing network protocol stack. We have added a hardware layer to capture energy-related optimizations that effect the processor and/or sensing unit. Of particular interest to this work, transmit power can be varied at the physical layer to manage transmission and interference ranges, sleep schedules and medium access schemes can be adjusted at the link layer, source rates can be managed at the transport layer and application-specific parameters can be tuned to manage data flows. To facilitate optimization of these "tuneable" parameters, battery life can be observed at the hardware layer, channel state information is available at the physical layer, link and end-to-end performance metrics can be measured at the link and transport layers and data flow characterization can be provided by the application layer.

A number of cross-layer protocols have been proposed in literature to exploit these opportunities. Common MAC/PHY layer solutions include scheduling at the MAC layer and power and/or rate control at the PHY layer [116],[117],[118]. Combined routing and MAC proposals include joint routing and scheduling [119],[120] and joint network coding and scheduling [121]. Transport cross-layer proposals include joint congestion control and scheduling with the MAC layer [122] and joint congestion control and power control in the PHY layer [123]. Cross-layer approaches also encompass three or more layers such as the joint routing, scheduling, power and rate control solution in [124].

Layer	"Tuneable" parameters	Provide control over….	Information available
Application	Data compression, data aggregation, application-specific parameters	Sensing coverage, generated data rate, application precision vs. accuracy?	Data flow characterization
Transport	Source rate		End-to-end throughput, end-to-end delay
Network/Routing	Routing matrix	Link utilization	
Link/MAC	Scheduling/access scheme, error detection/correction scheme, frame/packet size, radio sleep schedule	Transceiver energy consumption, available bandwidth, link delay	Frame/packet error rate, link level throughput, link level delay, contention delay
Physical	Transmit power, modulation and coding scheme (rate adaptation)	Transmission and associated interference range, transmission rate, bit error rate, transmitter power consumption	Signal-to-noise ratio (SNR), bit error rate (BER), other channel state information (CSI)
Hardware	Node shutdown/startup, variable CPU clock cycle, voltage scaling	Energy consumption by CPU and sensors	Battery life, computational delay

Table 3. Examples of typical cross-layer optimization parameters and information.

With the extensive work in the direction of cross-layer solutions, a number of researchers have begun to propose common frameworks [123],[125],[126],[127]. By formalizing the cross-layer discussion, these contributions offer a framework within which to provide analytic rigor to the development of proposed cross-layer solutions. Of particular note is the concept of "layering as optimization decomposition" proposed in [123],[127]. The fundamental approach of this work is that the overall network can be modeled as a global optimization problem and each layer in the resulting protocol stack can be viewed as a decomposed sub-problem with the coordinating variables serving as interfaces between the layers. From this perspective, different decompositions lead to different layering architectures, which can then be compared. The approach also complements the notions of both vertical decomposition into functional modules and horizontal decomposition across nodes to support distributed solutions as seen in Figure 13. To realize vertical decomposition, practical mechanisms such as the operating agent

in [126], which serves as a storehouse for global (cross-layer) variables must be implemented. Horizontal decomposition, meanwhile, typically implies some form of message passing between nodes.

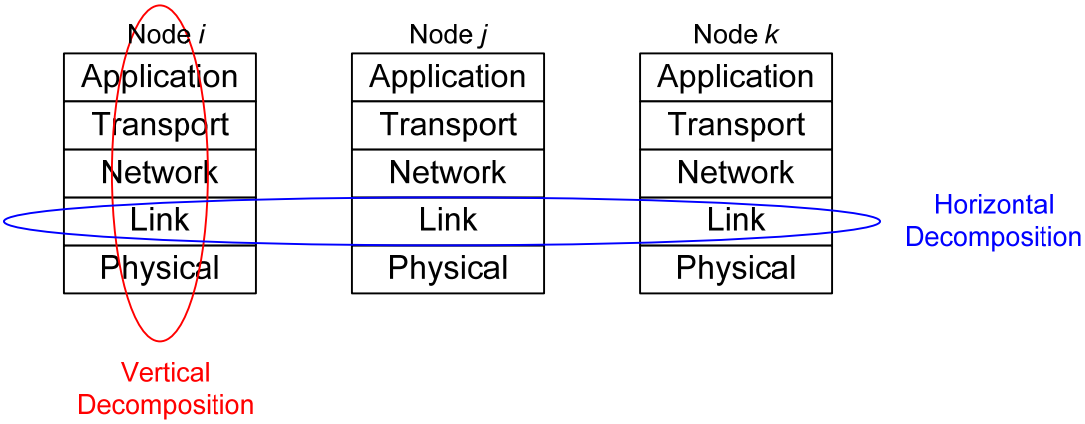


Figure 13. Cross-layer design: vertical vs. horizontal decomposition. Vertical decomposition requires a mechanism to share cross-layer information while horizontal decomposition typically implies some form of message passing.

C. MEDIUM ACCESS CONTROL PROTOCOLS

The wireless medium is a shared, broadcast medium and requires a mechanism to mediate access. As discussed in earlier, a wireless channel is time varying and asymmetric due to multipath propagation and fading. Accordingly, the wireless medium can be thought of to be comprised of a set of half-duplex links. These links are error-prone and particularly susceptible to burst errors [61]. A wireless network typically has no clear network boundaries and faces dynamic topologies due to node mobility, node state, and channel state. This section highlights the significant contributions to wireless medium access that exist in literature.

1. Contention-based Wireless Medium Access Control Protocols

We begin by discussing the landmark contention-based protocols in wireless communications. These have been proposed, by and large, to solve the medium access problem in wireless networks or, in some cases, specifically MANETs. They all have application to WSNs and, at a minimum, serve as the foundation of medium access research in WSNs.

a. ALOHA, Slotted ALOHA

ALOHA [20] was the first proposed wireless medium access scheme. It was developed at the University of Hawaii to interconnect the satellite campuses located throughout the island chain to the main campus on Hawaii. A simple scheme, a node immediately transmits a packet upon arrival. If a negative acknowledgement is received or no acknowledgement is received at the end of a predetermined timeout period, the packet is retransmitted after a random delay. It is important to note that this approach requires no synchronization or prior coordination. It is fully distributed in the sense that no coordination is required between nodes and each node makes an independent transmission decision.

The performance for ALOHA can be derived by calculating the probability of a successful transmission in which no collision occurs. The analysis assumes that the packets are of fixed size and follow a Poisson arrival process. Packet losses due to channel errors are assumed to be negligible. The vulnerability window (i.e., the window in which the potential exists for packets to be transmitted that will collide with the packet under consideration) for ALOHA is twice the packet transmission time and the probability of a successful transmission is the probability that no packets will arrive during this period. At steady state, the normalized throughput, ρ, for ALOHA can be shown to be

$$\rho = \alpha e^{-2\alpha} \tag{40}$$

where α is the total normalized offered load, which is defined as the total arrival rate (both new arrivals and retransmitted packets) normalized by the channel rate. The maximum normalized throughput (also referred to as the *capacity*) can be shown to be 0.184 at an offered load of 0.5 erhlangs.

Slotted ALOHA [64] achieved a 100% improvement in throughput over basic ALOHA by dividing the transmission time into slots and only allowing nodes to transmit at the beginning of a slot. At a cost of synchronization, this reduces the vulnerability window by half and improves the normalized throughput to

$$\rho = \alpha e^{-\alpha} . \tag{41}$$

Thus, for slotted ALOHA, the maximum normalized throughput can be shown to be 0.368 at an offered load of 1 erhlang.

b. Carrier Sense Multiple Access (CSMA)

Throughput for the ALOHA schemes is low because they do not take advantage of the fact that the propagation time is typically very small when compared to the packet transmission time. To take advantage of this observation, Leonard Kleinrock and Fouad Tobagi introduced the concept of carrier sensing and carrier sense multiple access (CSMA) [21]. The fundamental idea is that a node will sense the medium prior to packet transmission. If the medium is busy, the node will defer packet transmission to a later time. Thus, for CSMA, the vulnerability window is now based on the propagation time vice the transmission time as in the ALOHA schemes. Accordingly, the CSMA approaches show substantial improvement over ALOHA. This improvement, though, predictably degrades as the propagation time (or, equivalently, the propagation distance) increases.

In their landmark paper [21], Kleinrock and Tobagi offer several CSMA variants that differ according to how they defer when the medium is found to be busy. For nonpersistent CSMA, the node will wait a random delay before reattempting transmission. While the nonpersistent approach works well in reducing collisions, it results in wasted capacity, particularly at lower traffic loads. In *p*-persistent CSMA schemes, the node will transmit with some probability *p* upon hearing the medium go idle. For example, in a 1-persistent scheme, a node will defer upon finding the medium busy and will then transmit with probability of 1 (always) once the medium goes idle. This clearly removes the wasted capacity, but results in a higher probability of collision, particularly as the traffic load increases. The goal, then, in selecting the appropriate value for *p* is to manage the tradeoff between wasted capacity at lower traffic loads and increased collisions (and increased retransmissions leading to instability) at higher traffic loads. Slotted versions of these schemes are also proposed in [21], which further improve throughput performance. A comparison of the throughput performance of these contention-based schemes is provided in Table 4 and Figure 14. The parameter *a* in the

throughput equations for CSMA captures the impact of propagation delay on throughput. It is referred to as the normalized propagation delay and is defined as

$$a = \frac{\tau}{T_p} \tag{42}$$

where τ is the maximum propagation delay encountered by the network and T_p is the packet transmission time. As expected, it can be seen in Figure 15 that the capacity of all CSMA schemes degrades as the propagation time increases relative to the transmission time.

	Throughput, *S*	**Capacity (for *a* = 0.01)**
Pure ALOHA [20]	$\alpha e^{-2\alpha}$	0.184
Slotted ALOHA [64]	$\alpha e^{-\alpha}$	0.368
Unslotted 1-persistent CSMA [21]	$\frac{\alpha\left(1+\alpha+a\alpha(1+\alpha+a\alpha/2)\right)e^{-\alpha(1+2a)}}{\alpha(1+2a)-\left(1-e^{a\alpha}\right)+\left(1+a\alpha\right)e^{-\alpha(1+a)}}$	0.529
Slotted 1-persistent CSMA [21]	$\frac{\alpha\left(1+a-e^{-a\alpha}\right)e^{-\alpha(1+a)}}{(1+a)\left(1-e^{-a\alpha}\right)+ae^{-\alpha(1+a)}}$	0.531
Unslotted nonpersistent CSMA [21]	$\frac{\alpha e^{-a\alpha}}{\alpha(1+2a)+e^{-a\alpha}}$	0.815
Slotted nonpersistent CSMA [21]	$\frac{a\alpha e^{-a\alpha}}{1-e^{-a\alpha}+a}$	0.857
Perfect Scheduling		1.000

Table 4. Throughput performance for ALOHA and CSMA with a = 0.01

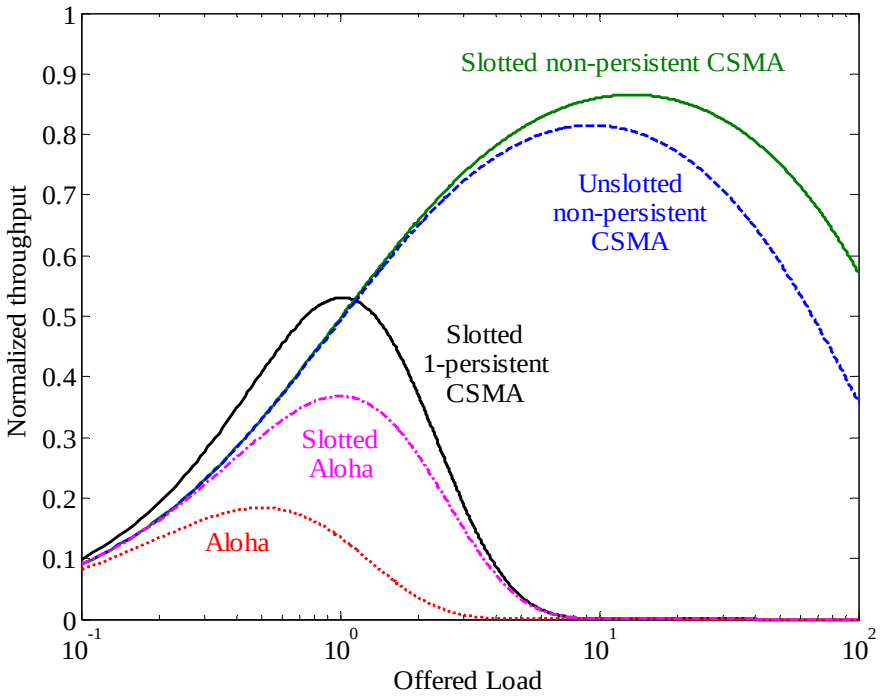


Figure 14. Comparison of throughput performance of CSMA and ALOHA as a function of offered load (After [21]).

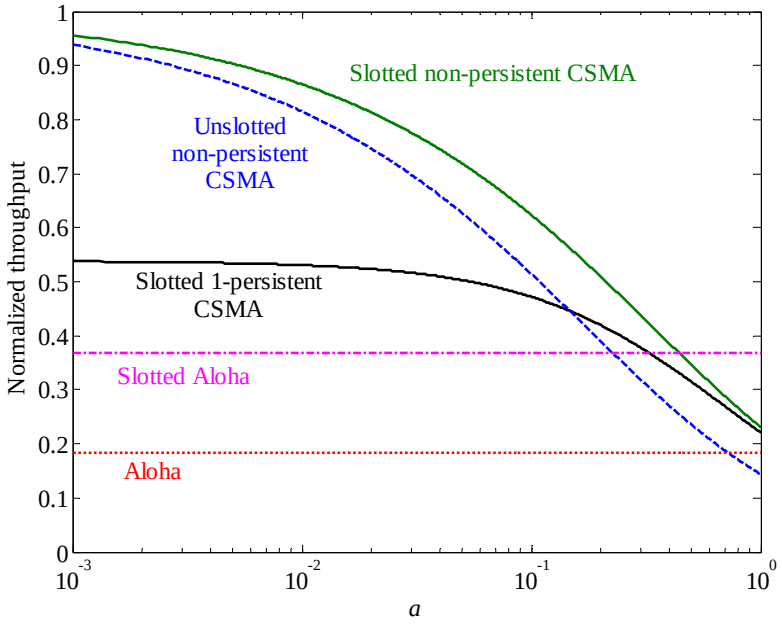
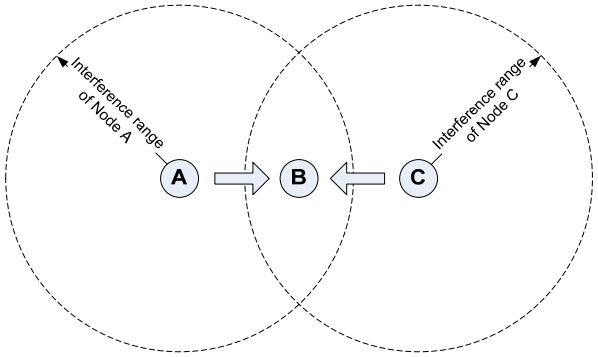


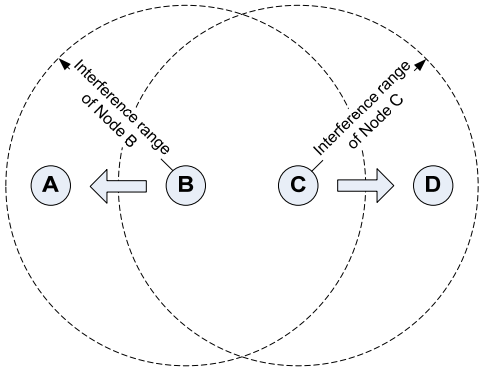
Figure 15. Comparison of throughput performance of CSMA and ALOHA for increasing values of *a* (After [21]).

c. MACA and MACAW

ALOHA and CSMA schemes suffer from the hidden node and exposed node problems and, accordingly, the actual performance is lower than the theoretical result documented in the previous section. The hidden node (alternately, the hidden terminal) problem [66] occurs when two nodes are within "hearing" range of a third node but not within "hearing" range of each other. Because they are out of range of each other, they can both transmit simultaneously, causing a collision at the third node. This can be seen in Figure 16(a). In the exposed node problem [67], adjacent nodes are within hearing range of each other and therefore are prevented from transmitting to a third node, even if it is not within hearing range of both nodes. This can be seen in Figure 16(b).



(a)



(b)

Figure 16. (a) The hidden node problem. Nodes A and C are out of range of each other, but both in range of node B. Nodes A and C can transmit simultaneously and cause a collision at B. (b) The exposed node problem. Although node D is out of range of the transmissions of B and node A is out of the range of the transmissions of node C, concurrent transmission of B to A and C to D is prevented (After [111]).

The first single channel solution to the hidden node and exposed node problems was MACA [67]. In MACA, a data exchange is preceded by a control exchange on the same channel. The control exchange is comprised of a Request-to-Send (RTS) packet sent by the sender to the receiver followed by a Clear-to-Send (CTS) packet sent from the receiver to the sender. This control and data exchange is illustrated in Figure 17(a). These control packets are assumed small relative to the data packets and they both contain the amount of data to be sent, which allows neighboring nodes to estimate the length of the upcoming data transmission. Upon hearing an RTS transmission, a neighboring node defers long enough to allow a CTS response to be sent. Upon overhearing the corresponding CTS, the station will defer for the length of time indicated by the amount of data to be transmitted. If no CTS is heard, the neighboring node assumes the destination is out of range and it is once again clear to transmit. This overcomes the exposed node problem. If a neighboring node overhears a CTS but not the initiating RTS, it assumes the receiver is in range even though the sender is not. Again, it will defer for the length of time indicated by the amount of data to be transmitted. This overcomes the hidden node problem. A critical assumption in this work is that the transmission and reception ranges for each station are roughly equivalent. MACA eliminates carrier sensing at the sender but retains the collision avoidance at the receiver. The major benefit of this approach is that it significantly reduces (but does not completely eliminate) the probability of data packet collisions. Because the control packets are smaller, the probability of control packet collisions is reduced and the cost of the collision is also reduced in terms of wasted medium access time.

MACAW [128] improved the performance of MACA over error-prone wireless links by adding an acknowledgement packet to the control exchange as seen in Figure 17(b). This allows for link level recovery from lost packets vice transport layer recovery. Simulation results indicate that in the presence of no packet losses, the throughput overhead of the ACK packet is approximately 9%. In the presence of packet losses, MACAW outperforms MACA as the loss ratio rises above 0.001. At a loss ratio of 0.01, the throughput of MACAW is shown to be roughly twice that of MACA.

Floor Acquisition Multiple Access – Non-persistent Transmit Request (FAMA-NTR) [129] introduced a non-persistent carrier sensing scheme into the RTS/CTS exchange. This combines collision avoidance both at the receiver (as in MACA [67] and MACAW [128]) and at the sender (as in CSMA [21]). The family of FAMA protocols (of which MACA and MACAW can be thought of as variants) acquire control of the channel (called the floor) prior to data transmissions to eliminate data packet collisions. MACA can be thought of as an ALOHA version of FAMA that does not require carrier sensing prior to transmission of the RTS (i.e., the RTS packet is transmitted upon arrival as in ALOHA). Slotted versions of MACA and FAMA-NTR are also presented in [129] as well as throughput analysis based on the CSMA throughput analysis of [21].

Other follow-ons include MACA—By Invitation (MACA-BI) [130] in which the RTS portion of the RTS/CTS exchange is suppressed, a traffic prediction algorithm is employed to predict when neighbors have traffic to send and the receiver initiates the transmission through Ready-to-Receive (RTR) packets, as well as Multiple Access with Reduced Handshake (MARCH) [131], which reduces the overhead of the RTS/CTS exchange over multiple hops by taking advantage of overheard CTS packets to eliminate redundant RTS transmissions. Figure 17 provides a good comparison of MACA, MACAW, and MACA-BI collision avoidance techniques.

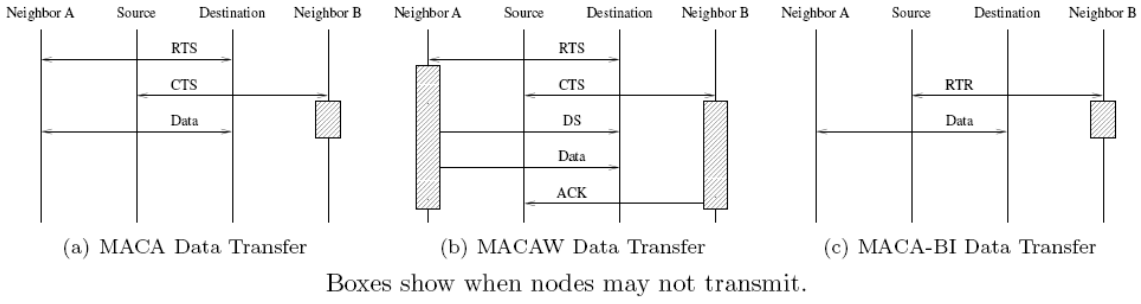


Figure 17. Comparison of collision avoidance techniques proposed in MACA, MACAW, and MACA-BI (From [132]).

d. IEEE 802.11 Medium Access

The medium access specification in the IEEE 802.11 Standard [24] is comprised of three components: the distributed coordination function (DCF), the point coordination function (PCF), and the hybrid coordination function (HCF). DCF is a fully distributed, contention-based access scheme while PCF is a centralized, contention-free approach that is laid on top of DCF. HCF combines the functionality of both DCF and PCF to support QoS-capable stations. In this subsection, we will focus on DCF, but include a brief discussion of PCF and HCF to understand how the contention-free mechanism is overlaid on top of the contention-based foundation.

DCF utilizes carrier sense multiple access with collision avoidance (CSMA/CA), which combines the mechanisms proposed in [21], [67], [128], and [129]. A station wishing to transmit a packet senses the medium. If the medium is idle for the duration of a specified minimum time period, the station will begin transmission. If the medium is busy or becomes busy before the specified time period is complete, the station will defer until the end of the current transmission. When the transmission completes and the medium becomes idle, the station will back-off a random length of time and then reattempt transmission. This can be seen as a version of nonpersistent CSMA [21]. To prevent a single station from unfairly monopolizing the medium, a station also defers prior to a transmission attempt that immediately follows a successful transmission. The back-off period is defined by a random number of fixed-size slots and the associated counter is decremented only while the medium remains idle (i.e., the countdown is suspended while the medium is busy). A positive acknowledgement (ACK) packet is included to confirm successful packet delivery as in MACAW [128]. In an optional extension, RTS/CTS packets may be used as in MACA [67] to acquire the channel and reduce the probability of data packet collisions.

Carrier sensing is accomplished through both a physical mechanism (detailed in the PHY specification of the standard) and a virtual mechanism. The virtual mechanism makes use of the network allocation vector (NAV), which is a counter that maintains the state of the medium. The NAV is updated based on message duration fields

contained in overheard RTS and CTS packets and counts down at a uniform rate. The medium is considered idle when no transmission is detected by the physical carrier sensing mechanism and the NAV is zero.

The length of the random backoff time in slots is determined by the selection of a random integer from the uniformly distributed interval [0,CW] where CW is known as the contention window and its size is bounded by the parameters CW_{min} and CW_{max}. Initially, CW is set to CW_{min}. With each subsequent unsuccessful transmission attempt, the size of the contention window exponentially increases (i.e., doubles) up to the maximum value of CW_{max}. At the completion of a successful transmission, the size of the contention window is reset to CW_{min}.

Prioritized access to the medium is provided by a mechanism called the interframe space (IFS). The IFS is the period that the medium must be idle before the station can either transmit or begin/resume the backoff process. Thus, a station with a shorter IFS can seize the medium before a station with a longer IFS and will have priority over that station. Table 5 contains a list of the IFSs used by DCF and PCF in relative order from shortest to longest. An overview of the DCF access method can be seen in Figure 18. PCF provides centralized, contention-free access to the medium through the use of a polling mechanism. The polling master is known as the point coordinator and resides at the access point. PCF is overlaid on top of DCF and priority access to the medium is provided to the point coordinator by the PCF IFS (PIFS) mechanism.

SIFS	Short interframe space	Used for ACK frames, CTS frames, and PCF poll responses
PIFS	PCF interframe space	Used by PCF to gain access to medium
DIFS	DCF interframe space	Used for DCF frames (excluding ACK and CTS)
EIFS	Extended interframe space	Used for resynchronization (triggered by PHY after reception of erroneous MAC frame)

Table 5. IFSs used in DCF and PCF.

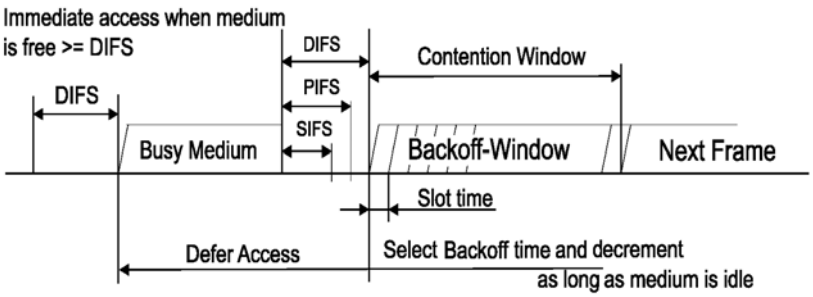


Figure 18. Basic DCF access scheme for IEEE 802.11 standard (From [24]).

The primary shortfalls of the IEEE 802.11 protocol when applied to wireless sensor networks are that (1) it is not designed for a multi-hop wireless network [133] and (2) it results in high energy consumption when nodes are idle [22]. The latter is primarily due to the idle listening that comes from the requirement to monitor the channel to be to be available to receive packets. Even in the case of an ongoing transmission destined for another node, a node with packets queued must monitor the channel to determine when it becomes free. IEEE 802.11 does include a power-save mode [24] (discussed earlier) that allow nodes to periodically sleep, but it is designed for a single-hop network and requires the use of a central controller (in this case, the access point) to coordinate sleep cycles and buffer messages that arrive when nodes are sleeping. Accordingly, this power-save mode does not scale well and is not suitable for multi-hop sensor networks [22].

2. Contention-free Wireless Medium Access Control Protocols

In contrast to contention-based approaches, the goal of contention-free access schemes is to completely eliminate collisions. This is a subtle point in that, in practice, a scheme is often considered contention-free if it eliminates only data packet collisions. Contention-free access schemes can be approached by asking two fundamental questions: “How is the medium divided?” and “How are the available slots scheduled?”

The wireless medium has been divided by time, frequency, code, and space. Respectively, these are classified as time division multiple access (TDMA), frequency division multiple access (FDMA), code division multiple access (CDMA), and space division multiple access (SDMA). While FDMA is still in use, TDMA has become more

prevalent in satellite systems due to the effectiveness of the digital techniques (including error correction) and the lack of intermodulation noise [61]. Meanwhile, 3G cellular technology predominately utilizes CDMA because it is less susceptible to multipath effects and, rather than a fixed bound of users as in FDMA and TDMA, CDMA performance drops off gradually as the number of users increases [61]. Multiple antenna approaches such as Multiple Input Multiple Output (MIMO) employ SDMA by spatially separating the transmissions [10]. While FDMA, CDMA, and SDMA all allow multiple simultaneous transmissions, both FDMA and SDMA imply multiple channels, which add to the complexity and hardware requirements at both the sender and the receiver. This work is focused on single channel/single antenna medium access solutions and, accordingly, we will not cover FDMA or SDMA. Additionally, due to the high computational complexity of CDMA schemes [30], our attention will be limited to TDMA.

Scheduling can be either fixed (fixed assignment multiple access – FAMA) or dynamic (demand assignment multiple access – DAMA) and the assignment can either be accomplished by a centralized controller or in a distributed fashion. Fixed scheduling is not responsive to changes in network topology and typically results in efficient utilization of the medium in a dynamic network environment [61]. The caveat is that the difference between a fixed versus dynamic assignment is only an issue of time-scale. By this, we mean that a dynamic assignment algorithm can be viewed as fixed if we examine it over a short enough time period. This leads to the tradeoff inherent in choosing the frequency of schedule updates in a dynamic scheme: the more frequent the updates, the more responsive the protocol to topologies changes, but the more overhead incurred. Thus, it is not always clear (or true) that more dynamic assignment schemes will be more efficient. Additionally, although centralized approaches are common and often highly efficient, they face inherent scalability issues. It should be noted, though, that a number of cluster-based approaches have been proposed in literature to address the scalability problem in a centralized approach [89],[134].

3. Wireless Sensor Network Medium Access Control Protocols

The fundamental metric (or objective) in medium access control for WSNs is typically node and/or network lifetime [30]. Due to the envisioned large-scale deployments, scalability is another important metric. Secondary metrics include latency, throughput, and utilization while fairness is rarely addressed. Numerous protocols have been proposed in literature and it would be unrealistic to try to discuss them all here. This section will survey a representative set of some of the more well accepted and well-studied proposals. Because we are interested in a fully distributed, peer-to-peer medium access solution, we will not address cluster-based approaches such as the Low-Energy adaptive Clustering Hierarchy (LEACH) protocol [89] and the Group TDMA protocol [134], which require the overhead of cluster establishment and maintenance. We will also not discuss multi-channel solutions such as the Power Aware Multi-Access with Signaling (PAMAS) [96].

a. IEEE 802.15.4

The IEEE 802.15.4 standard [25],[135] provides low-rate, low-power medium access for wireless personal area networks (PANs). Data rates of 20, 40, 250, and 851 kb/s are available in the 868, 915, and 2450 MHz frequency bands and three ultra-wideband (UWB) frequency bands at 500 MHz and 3.1 GHz to 10.6 GHz. Operation is supported for both star and peer-to-peer topologies although both require the use of a central PAN controller as shown in Figure 19. For non-UWB operation, channel access is provided through the use of slotted and unslotted CSMA/CA depending on whether the protocol is in the synchronized beacon mode or the unsynchronized mode, respectively. UWB channel access is provided through the use of ALOHA. In the beacon mode, synchronization beacons are transmitted by the PAN and bound a superframe (shown in Figure 20) that includes both an active period and an inactive period. The active period can include both a contention-based and contention-free period, which is controlled by the PAN. Nodes are allowed to sleep during the inactive period. Three data transfer modes are identified in the standard: uplink, downlink, and peer-to-peer. Uplink and downlink modes are specified for both beacon enabled and non-beacon enabled networks. The peer-to-peer mode is required only in the case of the peer-to-peer topology

and is not specified in the standard. IEEE 802.15.4 is the default communication protocol included on the latest generation of MICA [37] and TELOS [34] motes.

The requirement for a PAN controller leads to scalability concerns and is the primary shortfall when applying IEEE 802.15.4 to large-scale wireless sensor networks [132]. Additionally, although cluster-tree formations are mentioned in the standard and analyzed in [136], most published work focuses on the star topology [137],[138],[139],[140] and the peer-to-peer topology remains largely unexplored.

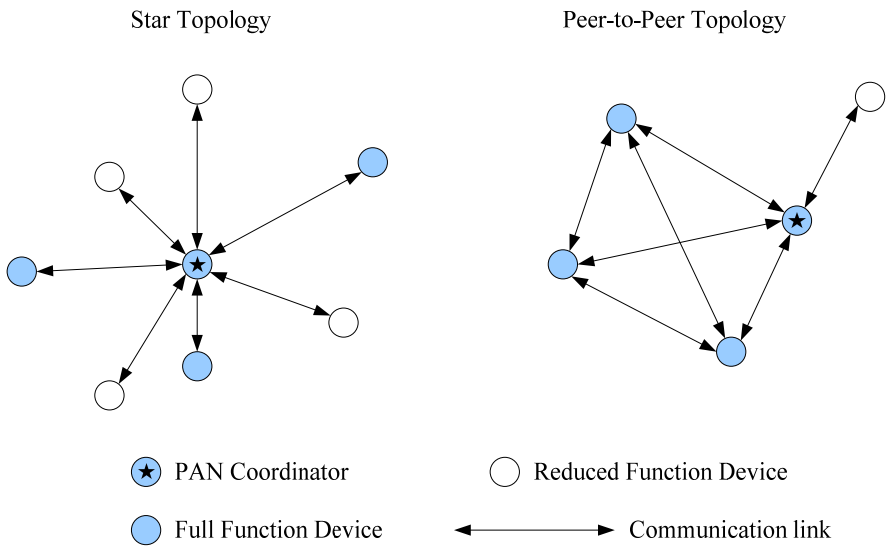


Figure 19. Topologies supported by the IEEE 802.15.4 (From [25]).

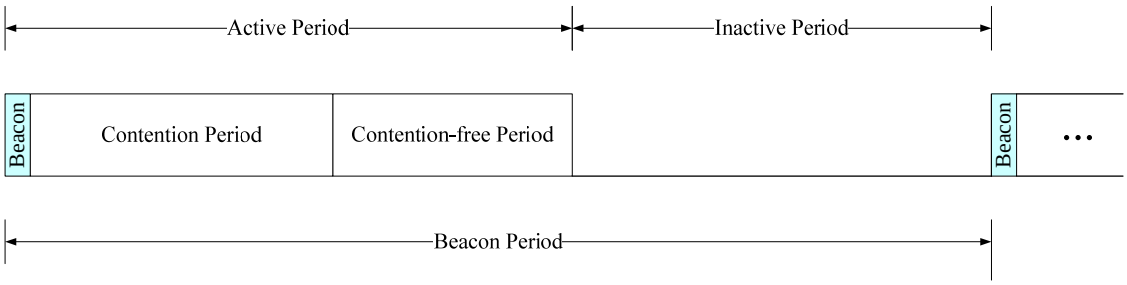


Figure 20. Beacon-enabled superframe format in IEEE 802.15.4 (After [132]).

b. Contention-based Access: S-MAC, B-MAC

A contention-based medium access scheme, S-MAC [22] is probably the most often referenced and studied MAC protocol for wireless sensor networks. It is an IEEE 802.11-based protocol where energy savings is achieved by establishing coordinated sleep and listening periods. Nodes wakeup during the listening periods to see if they have any inbound traffic. If so, they stay awake to receive traffic. If not, they return to sleep. The sleep and listen periods are defined ahead of time, do not change and are coordinated through synchronization within virtual clusters. As shown in Figure 21, listen periods begin with a synchronization period in which nodes transmit SYNC packets. Carrier sensing and RTS/CTS packets are used to reduce the probability of data packet collisions. S-MAC also makes use of message passing in which long messages are broken into frames and transmitted via bursts.

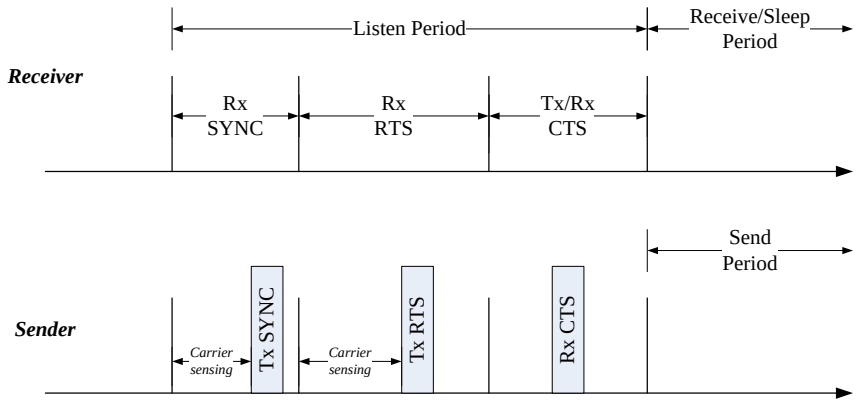


Figure 21. S-MAC operation (From [30]).

The authors of [22] identify sleep delay as the packet latency introduced by the sleep periods. This problem is compounded for multi-hop paths and an adaptive listening technique is included. Nodes that overhear transmission to a neighbor then wake up following the transmission in the event that the traffic will be forwarded onto them (the RTS/CTS packets contain the message duration). Thus, the neighbor does not need to wait for the next scheduled listen period and the data can be forwarded immediately.

Shortfalls in S-MAC include the fixed sleep and listen cycle, which leads to poor performance in varying loads. Timeout-MAC (T-MAC) [102] attempts to address

this by dynamically terminating the listening period when the length of time that no "activation event" has occurred exceeds a predetermined threshold. While this approach results in an adaptive sleep cycle, it also leads to the "early sleeping" problem. Several solutions to this problem are provided in [102]. T-MAC is shown to provide better performance over varying loads than S-MAC.

S-MAC also suffers from compounded sleep delay. Dynamic Sensor-MAC (DSMAC) [141] addresses this by implementing a dynamic sleep/listen cycle. A receiver doubles the duty cycle when sleep latencies get high (doubling maintains virtual cluster synchronization while dynamically increasing the duty cycle). Latencies are broadcast during the sync period and the sender doubles its duty cycle provided its battery threshold is high enough. DSMAC is shown to provide reduced latency and lower average power consumption per packet.

The Berkeley Media Access Control (B-MAC) protocol [65] is a lightweight alternate to S-MAC. It is comprised of four medium access mechanisms that include clear channel assessment (CCA), backoffs, link layer acknowledgements and low power listening (LPL). Each mechanism is provided with interfaces that allow network services to turn it on or off as well as adjust its functionality. B-MAC does not have the synchronization requirements of S-MAC and is approximately 72% smaller than S-MAC when an RTS/CTS and message fragmentation service is provided to mirror the functionality of S-MAC. B-MAC throughput is approximately 2 times better than that of S-MAC, but converges to S-MAC as contention increases. In multi-hop simulations, B-MAC demonstrates improved latency over S-MAC running with a 10% duty cycle. S-MAC power consumption is better than B-MAC, but B-MAC is within 25% of S-MAC.

c. Contention-free Access: TRAMA, LMAC

The operation of TRAMA [19], a well-studied TDMA-based medium access protocol for wireless sensor networks, is described in detail in Section II.D.3.b. Simulation results provided in [19] demonstrate that due to its schedule-based approach, TRAMA provides better packet delivery rates than CSMA, IEEE 802.11, and S-MAC (approximately 40% better than S-MAC and CSMA and 20% better than IEEE 802.11). This comes at the cost of increased latency due to the TDMA schedule. Additionally,

while S-MAC provides a higher percentage of sleep time than TRAMA, the average length of a sleep interval is greater in TRAMA. This highlights the need to balance total sleep time with the average duration of a sleep interval to account for the additional energy cost of node wakeup and shutdown. FLAMA [74] is a tree-based follow-on to TRAMA and is also discussed in Section II.D.3.b. In simulation and testbed experiments, FLAMA provides better latency performance and a larger percentage of sleep time than TRAMA. Its percentage of sleep time is comparable to or better than S-MAC. FLAMA, however, is not suitable for traffic-flows that do not adhere to a tree-based structure.

The authors of [119],[142],[143],[144] have proposed a series of TDMA-based, energy efficient protocols for wireless sensor networks that are based on a TDMA scheme in which the time slot is subdivided into control and data phases. In the Lightweight Medium Access Protocol (LMAC) [142], each time slot is subdivided into Control Message (CM) and Data Message (DM) phases. The CM phase is used to announce the controller node ID, the intended receiver, the length of the data, the distance in hops to the gateway (used for routing) and provide synchronization by including the sequence number of the time slot. All nodes are required to listen in on the CM of all nodes in their neighborhood. The DM phase immediately follows the CM phase and nodes are allowed to sleep if they are neither the designated transmitter nor receiver. Because multiple nodes can pick the same slot (particularly during network initialization), a mechanism is provided in the CM for neighbors to announce that they have detected a collision. This allows the affected nodes to re-compete for an available slot. The authors of [142] do not discuss how collision detection is accomplished. The protocol also implements a simple least-hop routing solution to the gateway that selects the next node to be the node closest (based on hop count) to the gateway. LMAC claims to extend the lifetime of the network (as defined by a 30% threshold for the number of nodes that have expired) by a factor of 3.8 over S-MAC. The Adaptive, Information-centric and Lightweight MAC (AI-LMAC) [143] modifies LMAC by assuming a tree-based query application and provides per node priority proportional to the expected amount of transmitted data. No performance comparisons are made in [143].

Other TDMA-based proposals include the Flexible-Schedule-Based TDMA protocol (FlexiTP) [145], which assumes a tree-structure, and [146], which assumes a rectangular or hexagonal grid topology.

d. Hybrid Access: Z-MAC

Zebra MAC (Z-MAC) [23] is a hybrid medium access that attempts to take advantage of the improved performance of CSMA at low contention levels and the improved performance of TDMA at high contention levels. Unlike the work in this dissertation, though, Z-MAC treats the flows in aggregation and responds only to overall network contention levels. In Z-MAC, a TDMA frame structure with assigned time slots is used to assist in CSMA contention resolution.

Z-MAC initialization begins with a setup phase that includes two-hop neighborhood discovery, slot assignment using DRAND [68], a local frame exchange, and global time synchronization using a synchronization protocol such as TPSN [81]. The local frame exchange is used to establish the TDMA frame size. Rather than promulgating a common maximum slot number throughout the network, the authors propose a time frame rule that allows nodes to locally determine a "non-conflicting" frame size within their two-hop neighborhood. A node's local frame size is based on the maximum slot number within its two-hop neighborhood and is chosen as 2^k to satisfy

$$2^{k-1} \le S_{max} < 2^k - 1 \tag{43}$$

where k is an integer and S_{max} is the maximum slot number assigned in the two-hop neighborhood. Since the frame lengths are powers of two, it can be shown that slot assignments in overlapping neighborhoods will not conflict [68]. At the end of the setup phase, the slot assignments and frame sizes are distributed throughout the two-hop neighborhood. Due to the overhead associated with this setup phase, it is only executed at protocol startup and when the network topology changes significantly.

Z-MAC is implemented using the backoff, CCA and LPL interfaces of B-MAC [65] and consists of two modes: low contention level (LCL) and high contention level (HCL). In LCL, any node can compete for any slot while in HCL only slot owners and their one-hop neighbors can compete for a slot. In both modes, the slot owner has

priority and all nodes can compete for non-assigned slots. This transmission scheme is accomplished as follows. Prior to transmission, a node with queued data picks a random backoff within a specified contention window and then senses the medium when the backoff expires. If the medium is free, the node transmits, if not, the node waits until the medium is free and reattempts transmission. If a node is the current slot owner, the contention window is $[0, T_0]$. In LCL mode or in HCL mode when the slot is not owned by a two-hop neighbor, the contention window is $[T_0, T_1]$. If the slot is owned by a two-hop neighbor in HCL mode, the node defers until its own slot occurs or a slot occurs that is not owned by a two-hop neighbor.

A node is in HCL mode if it has received an explicit contention notification (ECN) message from one of its two-hop neighbors within a specified time period, T_{ECN}. Nodes make local decisions to send ECN messages based on the average number of backoffs per packet they are experiencing. This is a passive measure of the noise level of the channel and the authors demonstrate the correlation between the noise level and the two-hop neighborhood contention level experimentally. Since any node can potentially transmit in any slot to any one-hop neighbor, energy savings are achieved by allowing nodes to sleep using the preamble sampling technique. Finally, local synchronization is required to maintain slot boundaries and is achieved using a one-way timestamp message exchange that includes weighted averages and a "trust factor" to minimize the impact of inputs from nodes that have drifted significantly since the last synchronization update. The authors point out that in the absence of local synchronization, the protocol defaults to CSMA.

As anticipated, the data throughput of Z-MAC is shown experimentally to outperform the contention-based B-MAC protocol at medium to high contention levels, but B-MAC demonstrates slightly better performance at low contention levels due to the overhead of the Z-MAC congestion window for non-slot owners. While both Z-MAC and B-MAC are shown to consume less power than S-MAC for low-data rate applications, the Z-MAC performance is slightly worse than B-MAC due to the larger backoff windows and the need for periodic local time synchronization messages. At higher data

rates, Z-MAC outperforms B-MAC (up to 40% better energy efficiency) due to the improved contention resolution provided by the TDMA-like behavior in HCL.

The authors of [104] propose a sender and receiver-based scheduling scheme to improve the energy performance of Z-MAC by reducing (but not eliminating) the need for preamble sampling. In their work, the first slot of each TDMA frame is designated as a pre-schedule slot and nodes with traffic to send contend in this slot to broadcast their schedules, which include both destination address and data length. It is unclear how this proposed scheme would perform under high contention conditions when a large number of nodes have data to transmit and it becomes increasingly difficult for nodes to attain the medium for schedule transmission in the pre-schedule slot.

PQ-MAC [147] adds a prioritized medium access scheme to the Z-MAC structure by ordering the contention period at the beginning of every slot based on priority. The slot structure for PQ-MAC is shown in Figure 22 where the contention period (CP) includes three levels of priority (Q, H, and L in order of decreasing priority). Based on packet priority, a slot owner can contend in the periods *T0*, *T2*, and *T4* while non-owners can compete for the slot in *T1*, *T3*, and *T5*. Multi-level queue support is provided and schedule broadcasts are also implemented using a superframe structure to improve energy efficiency. The throughput of PQ-MAC is shown to outperform S-MAC at high data rates. Both energy consumption and latency is shown to be better for PQ-MAC across a spectrum of data rates. No comparison is made between PQ-MAC and Z-MAC.

This chapter presented an overview of the body of work in the literature that serves as the launching point for the research effort of this dissertation. It covered important findings and proposals in wireless sensor network research and medium access control. Having provided this background, we now move onto the results of the dissertation research.

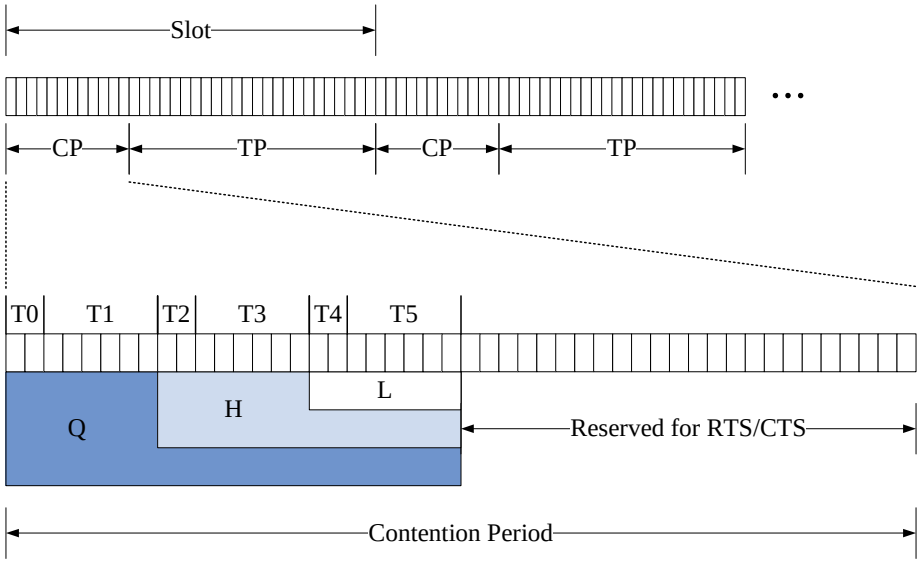


Figure 22. Slot structure of PQ-MAC (From [147]).

III. FLOW-SPECIFIC MEDIUM ACCESS

The centerpiece of the research effort of this dissertation is the proposal of a groundbreaking approach to wireless medium access that is rooted in the notion of providing medium access service on a per flow basis rather than in aggregation. Accordingly, we begin this work by introducing this novel medium access approach, contrasting it to traditional approaches and providing motivation for the work by demonstrating that it is capable of outperforming existing wireless medium access schemes. In subsequent chapters, we will present both traffic-adaptive and energy efficient realizations of this innovative medium access solution.

To lay the groundwork for our approach, this chapter begins by identifying the medium access requirements in a wireless sensor network. We then examine the delay performance of different medium access approaches and formally define the terms *flow-specific medium access* and *traffic-adaptive, flow-specific medium access*. We conclude by showing that traffic-adaptive, flow-specific medium access is capable of providing better delay performance than traditional contention-based and contention-free as well as hybrid approaches.

A. MEDIUM ACCESS REQUIREMENTS IN WIRELESS SENSOR NETWORKS

We begin the examination of the medium access requirements in a wireless sensor network by defining a *cooperative wireless sensor network* as one in which sensors exchange information to coordinate efforts and maximize application-related performance. In cooperative wireless sensor networks, traffic can be divided into multiple, distinct classes. As can be seen in applications such as wireless multimedia sensor networks [6], the data traffic is typically bandwidth-intensive but is tolerant to individual packet loss because the sensor data are correlated in both time and space. The associated control packet traffic, required to effectively utilize the capabilities of the dynamic sensor nodes, typically requires an end-to-end delay bound and is not tolerant to

losses but demands significantly less bandwidth. In general, for this traffic class, the packets are smaller and do not arrive as frequently but must be transmitted quickly and reliably.

An example of an unattended battlefield monitoring application of a cooperative wireless sensor network is shown in Figure 23. Here, the sensor network is comprised of a field of rotating video cameras. If the cameras have a 50 degree field of view, they will be able cover up to 93 m of target track at a distance of 100 m. As a high speed 60 mph (27 m/s) target passes through the field, camera rotation updates will occur on the order of once every second to keep the target centered and will be triggered by control packets sent from either the sink (the command and control node) or adjacent sensor nodes. These control packets need to be successfully transmitted despite the large data flow generated by the sensor field cameras. If each camera is capable of producing 320□240, 8-bit monochrome images at a frame rate of up to 20 fps, then the per node raw image data rate would be in excess of 12 Mbps.

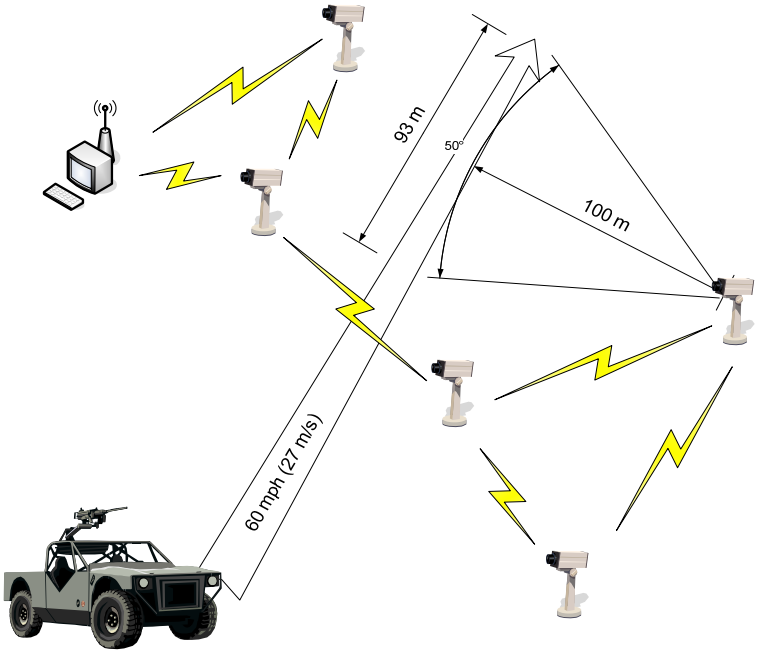


Figure 23. Unattended battlefield monitoring example of a wireless sensor network. The data traffic from the video cameras to the command and control point (sink) and the control traffic required to manipulate the camera (focus, azimuth, elevation, etc.) form two distinct traffic flows.

The proposed medium access control solution should meet the primary service requirements for both classes of traffic. Specifically, it should provide a high throughput solution to support the sensor data packets and a reliable solution with minimum end-to-end delay to support the control packet flow. Additionally, to provide robustness, it is desirable that the design be distributed so that each node is able to make a local transmission decision for each class of traffic. Finally, the protocol should support multi-hop networks in which all nodes are capable of handling all classes of traffic.

B. DELAY PERFORMANCE OF CONTENTION AND CONTENTION-FREE MEDIUM ACCESS

To provide further motivation for traffic-adaptive, flow-specific medium access, we now conduct an examination of the delay performance of various contention-based and contention-free medium access schemes. The mean packet delay for ALOHA [20], slotted ALOHA [64], several CSMA variants [21], and TDMA [26] is plotted in Figure 24 as a function of the normalized load. This normalized load is equivalent to the steady state throughput and is normalized by the channel rate. For the purposes of the plot, channel rate is 1 Mbps, packet size is 1000 bits, there are 100 slots in a TDMA frame (each slot is one packet length in duration) and $a = 0.01$ for the CSMA schemes. The CSMA plots represent the best case achievable delay at steady state. For these delay curves, we assume Poisson arrivals and the appropriate delay equations can be found in [20], [21], [26], [64].

It can be seen that at low loads, the delay performance of the contention schemes is better, while at higher loads, the delay performance of the non-contention scheme is better. It is natural then to ask if we can get the delay performance of CSMA at low loads and that of TDMA at high loads. This is precisely the strategy of hybrid approaches, such as [28],[69], which treat the flows in aggregate and transition from a contention-based approach to a non-contention-based approach as the load increases. In an aggregate flow that is comprised of both low and high demand flows, these hybrid schemes have the disadvantage of increased delay for flows that could take advantage of the lower delays

associated with the contention-based approaches. In contrast, we propose to treat each flow individually to optimize both the overall performance and the performance on a per flow basis.

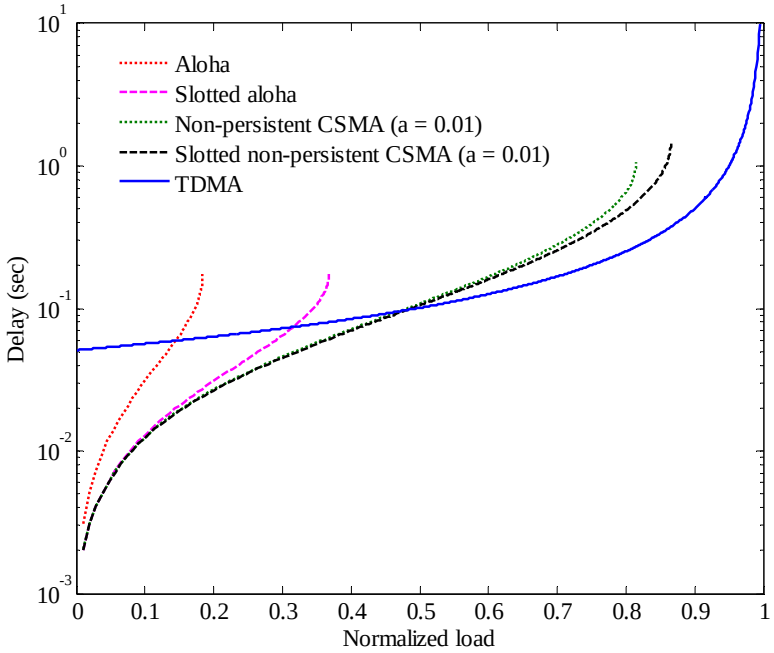


Figure 24. Packet delay plotted as a function of normalized load for TDMA and CSMA. Channel rate is 1 Mbps, packet size is 1000 bits, and there are 100 slots (one packet length in duration) in the TDMA frame.

C. TRAFFIC-ADAPTIVE, FLOW-SPECIFIC MEDIUM ACCESS

We now formally define the terms flow-specific medium access and traffic-adaptive, flow-specific medium access and provide an example to illustrate the concept.

Definition: *Flow-specific medium access control* is a medium access approach that provides medium access on a per flow basis. It is capable of concurrently providing different medium access schemes to different traffic flows.

Definition: *Traffic-adaptive, flow-specific medium access control* is a flow-specific medium access approach that is capable of dynamically switching between multiple medium access schemes to respond to traffic variations within a given flow.

As an illustrative example, we examine an aggregate flow that is comprised of two individual packet flows. We assume that the load of the first flow is low while the load of the second flow varies from low to high. The aggregate flow demand, then, will vary with the second flow. This example models the behavior of an event-based wireless sensor network (such as that discussed in the previous section and shown in Figure 23) that includes both a control flow to provide sensor coordination within the network and a data flow that corresponds to sensor data transmission to a designated sink. Prior to event detection, the demand of both flows is low (perhaps in a periodic reporting state). Upon event detection, the control flow remains relatively low demand (control packets are small in size and are only needed periodically to update sensor parameters) while the data flow will increase dramatically as recorded event data is forwarded to the sink.

In this example, contention-based [20],[21] and non-contention-based [26] schemes will treat the flows in aggregate and provide either contention-based or non-contention-based access, respectively, to the combined flow. A traffic-adaptive, hybrid scheme [28],[69] will again treat the flows together, but will transition from contention-based to non-contention-based medium access when the demand of the aggregate flow reaches some threshold. In contrast, a traffic-adaptive, flow-specific approach will treat the two flows individually by continuing to provide contention-based medium access to the low demand control flow while the data flow is transitioned from contention-based to non-contention-based access as its load increases.

D. DELAY PERFORMANCE OF FLOW-SPECIFIC MEDIUM ACCESS

Defining the aggregate delay performance as the weighted sum of the delay performance for the individual flows, we can evaluate and compare the delay performance of the different approaches for our two-flow example [21],[26],[28]. In Figure 25, we plot the mean aggregate packet delay as a function of aggregate load for

the four approaches. The normalized load of the first flow is fixed at 0.1 while the load of the second flow is allowed to vary from 0.0 to 0.8. We can clearly see that while the hybrid approach takes advantage of the lower delays of CSMA in the low contention region and TDMA in the high contention region, the traffic-adaptive, flow-specific approach offers better overall delay performance in the high contention region by allowing the low demand control flow to remain in the contention-based mode.

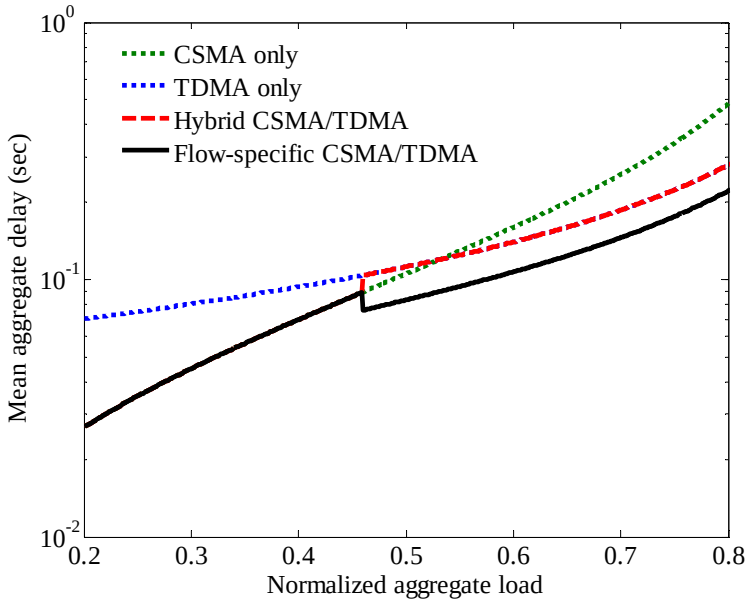


Figure 25. Packet delay plotted as a function of normalized load for slotted nonpersistent CSMA [21], TDMA [26], hybrid and flow-specific medium access (using CSMA/TDMA). Channel rate is 1 Mbps, packet size is 1000 bits, there are 100 slots in a TDMA frame (each slot is one packet length in duration) and $a = 0.01$ for the CSMA schemes. The CSMA plot assumes steady state and represents minimum achievable delay.

Figure 25 illustrates the advantage of a traffic-adaptive, flow-specific approach in this particular example. In the following theorem and associated corollary, we extend this to the general case and show that the traffic-adaptive, flow-specific approach outperforms contention, non-contention and aggregate hybrid medium access schemes provided that the per flow switchover point between the access modes is chosen correctly.

Theorem: Given a suitable switching point is chosen at which a flow will transition between medium access schemes, flow-specific medium access will provide as good or better delay performance than contention, non-contention, and hybrid medium access schemes.

Proof: First, let us consider the case of the contention-based medium access scheme. Without a loss of generality, we will assume that the mean packet delays D_i for the N individual flows i are ordered as in $D_1 \le D_2 \le \cdots \le D_m \le \cdots \le D_{N-1} \le D_N$. The switching point between access schemes is then chosen such that

$$\begin{aligned} D_i^c &\le D_i^{nc} \quad \text{for all } i = 1:m \\ D_i^c &> D_i^{nc} \quad \text{for all } i = (m+1):N \end{aligned} \tag{44}$$

where D_i^c is the contention-based access scheme delay for flow i and D_i^{nc} is the non-contention-based access scheme delay for flow i. The mean aggregate delay for the flow-specific access scheme is

$$D_{flow} = \sum_{i=1}^{N} \left(\frac{\lambda_i}{\lambda} \right) D_i \tag{45}$$

and the mean aggregate delay for the contention-based scheme is

$$D_{cont} = \sum_{i=1}^{N} \left(\frac{\lambda_i}{\lambda} \right) D_i^c \tag{46}$$

where λ_i is the arrival rate for flow i and the aggregate arrival rate λ is the sum of the individual flow arrival rates. From (44), the mean overall delay of (45) is equivalent to

$$D_{flow} = \sum_{i=1}^{m} \left(\frac{\lambda_i}{\lambda} \right) D_i^c + \sum_{i=m+1}^{N} \left(\frac{\lambda_i}{\lambda} \right) D_i^{nc}. \tag{47}$$

Using proof by contradiction, suppose that the contention-based medium access provides lower aggregate mean delay than the flow-specific scheme or $D_{flow} > D_{cont}$. Expanding these,

$$\sum_{i=1}^{m} \left(\frac{\lambda_i}{\lambda} \right) D_i^c + \sum_{i=m+1}^{N} \left(\frac{\lambda_i}{\lambda} \right) D_i^{nc} > \sum_{i=1}^{N} \left(\frac{\lambda_i}{\lambda} \right) D_i^c \tag{48}$$

Breaking apart the contention-based term on the right side of the inequality, we have

$$\sum_{i=1}^{m}\left(\frac{\lambda_i}{\lambda}\right)D_i^c + \sum_{i=m+1}^{N}\left(\frac{\lambda_i}{\lambda}\right)D_i^{nc} > \sum_{i=1}^{m}\left(\frac{\lambda_i}{\lambda}\right)D_i^c + \sum_{i=m+1}^{N}\left(\frac{\lambda_i}{\lambda}\right)D_i^c, \tag{49}$$

which can then be reduced to

$$\sum_{i=m+1}^{N}\left(\frac{\lambda_i}{\lambda}\right)D_i^{nc} > \sum_{i=m+1}^{N}\left(\frac{\lambda_i}{\lambda}\right)D_i^c. \tag{50}$$

This implies that

$$D_i^c < D_i^{nc} \quad \text{for some } i = (m+1):N, \tag{51}$$

which contradicts (44). Thus, $D_{flow} \le D_{cont}$ and flow-specific medium access will provide as good or better delay performance than a contention-based scheme. The non-contention case is proven in a similar manner. Finally, the hybrid scheme can be considered as either a contention scheme when the aggregate load is below the switching point or a non-contention scheme when it is above. Accordingly, it can be broken into two cases and is proved in a similar manner as well. *Q.E.D.*

Corollary: Given a suitable switching point is chosen at which a flow will transition between medium access schemes and that there exist at least two flows, which are in two different medium access modes, flow-specific medium access will provide better delay performance than contention, non-contention, and hybrid medium access schemes.

Proof: This corollary follows directly from the theorem since it can be shown that the equality in performance only occurs when *m* is either 1 or *N*. The constraint that there exists at least one flow in each of the contention and non-contention modes implies that $1 < m < N$ and, therefore, that the delay performance of the traffic-adaptive, flow-specific approach is strictly better than the other schemes. *Q.E.D.*

From this discussion, it is clear that the performance of a traffic-adaptive medium access scheme is tied to the selection of the switching point. Returning to our two-flow example, the impact of the selection of the switching point can be plainly seen in Figure

26 where we plot mean aggregate delay versus normalized aggregate load for four different switching points. When the switching threshold is too low, the flow-specific scheme transitions to the non-contention mode early and the delay performance at low contention levels suffers. When the switching point exceeds the optimum value, the scheme transitions late and performance in the high contention range is reduced.

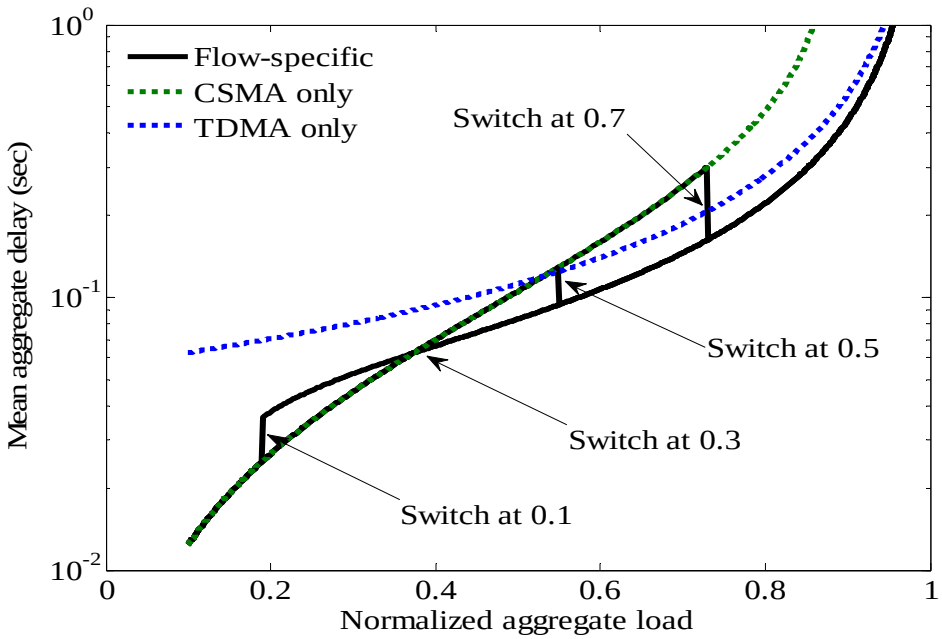


Figure 26. Flow-specific delay plotted against the normalized load and our compared to CSMA and TDMA for various switchover points. Channel rate is 1 Mbps, packet size is 1000 bits, there are 100 slots in a TDMA frame (each slot is one packet length in duration) and a = 0.01 for the CSMA schemes. The CSMA plot assumes steady state and represents minimum achievable delay.

In this chapter, we introduced the novel traffic-adaptive, flow-specific approach to medium access control. We examined the delay performance of various medium access approaches and showed that the novel traffic-adaptive, flow-specific medium access approach is capable of outperforming contention-based, contention-free and hybrid approaches provided a suitable switching point is chosen. In the next chapter, we present a flow-specific medium access scheme designed to realize this performance advantage and propose a queue-based, traffic-adaptive mechanism to dynamically implement the accompanying switching point.

THIS PAGE INTENTIONALLY LEFT BLANK

IV. TRAFFIC-ADAPTIVE COOPERATIVE WIRELESS SENSOR MEDIUM ACCESS CONTROL (CWS-MAC) PROTOCOL

In the previous chapter, we formally introduced the novel concept of traffic-adaptive, flow-specific medium access and provided theoretical performance results to demonstrate that it can potentially outperform existing contention and contention-free solutions. We now propose a traffic-adaptive, flow-specific implementation that realizes these performance advantages. To achieve this, we provide both a flow-specific medium access mechanism capable of delivering per flow medium access service and a traffic-adaptive mechanism capable of responding to changes in per flow load as well as overall network contention levels. The former represents the fundamental building block of the access scheme while the latter equates to effectively implementing the switching point we indentified in the previous chapter.

Accordingly, we begin by proposing the Cooperative Wireless Sensor Medium Access Control (CWS-MAC) protocol to meet need for flow-specific medium access and then offer a queue-based, traffic-adaptive mechanism to observe the traffic and trigger flow-specific changes in medium access service. By providing performance analysis for both the queue-based, traffic-adaptive mechanism in general and CWS-MAC in particular, we follow up on the ideal performance analysis of Chapter III with analysis of both a general queue-based traffic adaptive, flow-specific approach as well as the specific implementation of traffic-adaptive CWS-MAC.

This chapter begins with a description of CWS-MAC and then presents the queue-based, traffic-adaptive mechanism. A general performance model for traffic-adaptive, flow-specific medium access is then proposed and analyzed and delay and throughput performance analysis for traffic adaptive CWS-MAC is developed for all modes. The chapter concludes with simulation results that demonstrate the effectiveness of the proposed scheme and compare its performance to traditional CSMA and TDMA schemes.

A. COOPERATIVE WIRELESS SENSOR MEDIUM ACCESS CONTROL (CWS-MAC) PROTOCOL

The Cooperative Wireless Sensor Medium Access Control (CWS-MAC) protocol is designed to meet the high throughput requirements of the high bandwidth sensor data flow while providing priority access to the medium for the time-critical control packet flow. Priority is based on traffic flow rather than node identity and, therefore, requires that the medium access layer be application-aware. This flow priority is enforced globally across nodes rather than locally within each node (i.e., a node with a control packet has priority over another node with a data packet). Additionally, an acknowledgment-based reliability mechanism is included to support the loss-intolerant control traffic flow.

1. CWS-MAC Operation

CWS-MAC [18] is a fixed, flow-specific medium access control scheme that is designed to accommodate multiple flows based on flow demand. Application-aware, it combines the low demand delay performance of a contention-based scheme with the high demand throughput performance of a non-contention (scheduled) approach. An illustration of the CWS-MAC frame is provided in Figure 27. In [18], we refer to the "control" and "data" flows. In this dissertation, we generalize these and their respective medium access mechanisms to "contention-based" and "non-contention-based."

The underlying non-contention-based medium access mechanism is provided by a TDMA scheme in which nodes are assigned slots within the TDMA frame for transmission of their non-contention flow packets. Slot assignment can be accomplished through a dynamic, distributed scheduling algorithm such as [68], [71], [148] (as discussed in Chapter II).

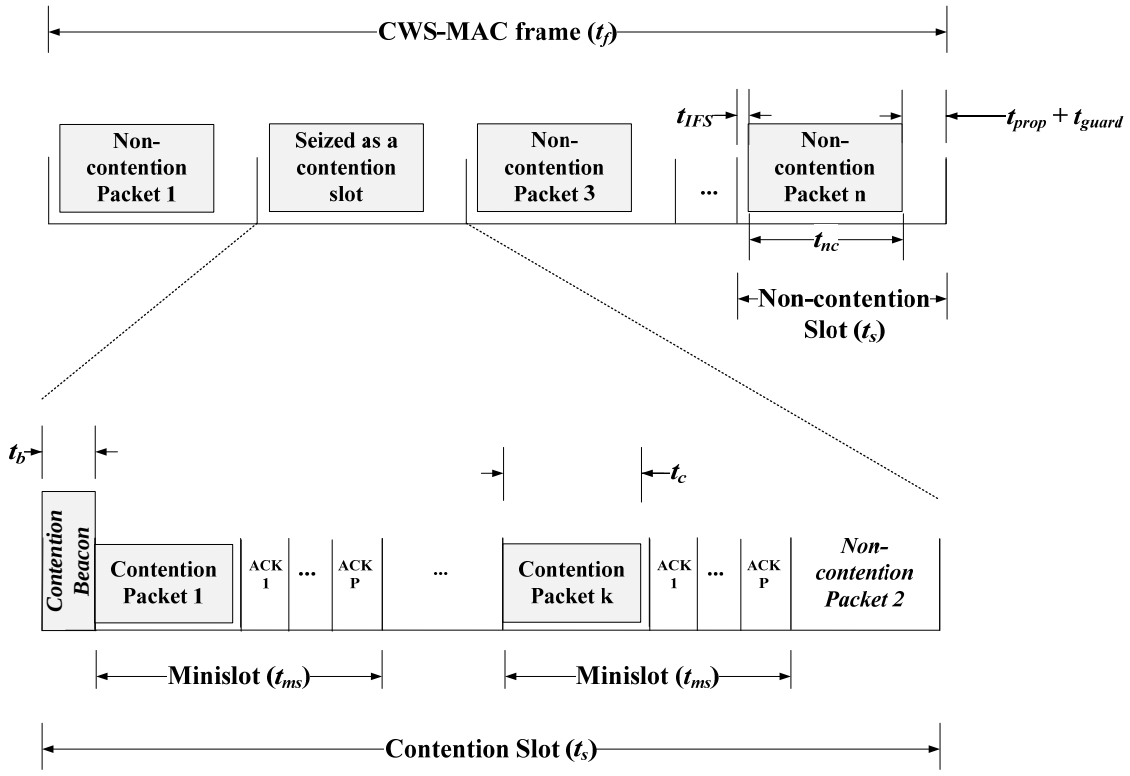


Figure 27. An illustration of the CWS-MAC frame.

The contention-based medium access mechanism is superimposed on top of the TDMA framing through the use of an interframe space and a contention beacon that effectively give the contention-based flow global (across node) priority over the non-contention-based flow. A node with contention flow packets to transmit signals its intent to seize the current TDMA slot by transmitting a contention beacon of length, t_b. Although not specifically addressed in [18], in a multi-hop network this beacon must be retransmitted to all two-hop neighbors of the originating node. A node with non-contention packets to transmit in its TDMA slot must wait for the duration of the interframe space, t_{IFS}, and then sense the medium. If the medium is free (i.e., no contention beacon has been transmitted in its two-hop neighborhood), the packet may transmit its non-contention packets. If a beacon is detected, the slot owner defers and the slot is effectively seized as a contention slot.

To ensure the non-contention flow is not "choked off," a portion of the original TDMA slot is set aside in the contention slot for use by the slot owner for non-contention

packet transmission. From a bandwidth allocation viewpoint, this reserved portion of the contention slot bounds the minimum allocation allotted to the non-contention mode. In general, the bandwidth allocation between the contention and non-contention modes in CWS-MAC is dynamic and responds to the contention packet arrival rate.

To reduce collisions among competing nodes with contention-based traffic to transmit within the two-hop neighborhood, the contention slot is subdivided into a series of transmission minislots. A version of slotted ALOHA [64], a node will transmit in a minislot with some predetermined probability (calculated as the inverse of the number of minislots in [18]) and an acknowledgement mechanism is included to recover from collisions.

Primary design parameters for CWS-MAC include the slot size, t_s, the minislot size, t_{ms}, the number of minislots, k, and the lengths of the control beacon and interframe space. We investigate these parameters in detail and provide a strategy for parameter selection in the following discussion.

2. CWS-MAC Timing Parameters

From a physical layer standpoint, the contention beacon must be long enough to be detected by the sensor nodes within communication range of the transmitting node. This work will not go into the specifics of the physical layer signal-to-noise ratio (SNR) and the resulting probability of detection calculations, but a related analysis can be found in [66].

The contention beacon length, t_b, is also constrained by its relationship to the interframe space period t_{IFS}. To ensure that a consistent value can be chosen for t_{IFS} across the network, it can be shown that the beacon reception period for the most distant node must overlap with that of the node closest to the transmitting node. This requirement results in the following bound on t_b:

$$t_b > \max\left(T_{prop}\right) - \min\left(T_{prop}\right) \tag{52}$$

where T_{prop} is the propagation time from the transmitting sensor node to the receiving sensor node. The interframe space value should then be selected to lie within this

overlapping time period. Conservatively assuming the minimum propagation time within the network to be zero, this condition can be shown to be equivalent to

$$t_b > t_{IFS} > t_{prop} \tag{53}$$

where $t_{prop} = \max(T_{prop})$ is defined as the maximum propagation time experienced by the network.

3. Slot Size and Slot Assignment

In CWS-MAC, the slot size is fixed and is governed by both the non-contention packet flow and the contention packet flow since a slot can act as either a non-contention slot or a contention slot. For a non-contention slot, the slot size, t_s, is bounded by

$$t_s \geq t_{IFS} + t_{nc} + t_{prop} + t_{guard} \tag{54}$$

where t_{nc} is the transmission time of a single non-contention packet and t_{guard} is an interval provided to accommodate timing synchronization errors between sensor nodes. For a contention slot, the slot size is bounded by

$$t_s \geq t_b + k\left(t_c + t_{ack} + 2t_{prop} + t_{guard}\right) \tag{55}$$

where k is the number of minislots in a contention slot, t_c is the transmission time of a single contention packet, and t_{ack} is the transmission time of a single acknowledgement packet. The inclusion of t_{prop} in (54) and (55) prevents a distant node from transmitting in its assigned time slot before the previous packet has cleared the network. This buffer is required due to the broadcast nature of the wireless medium.

By combining (54) and (55), we can arrive at a relationship between t_d and k. Selecting the equality in (54) and substituting into (55), we find that

$$\begin{aligned} t_{nc} \geq\ & t_b - t_{IFS} + k\left(t_c + t_{ack}\right) \\ & + \left(2k-1\right)t_{prop} + \left(k-1\right)t_{guard}. \end{aligned} \tag{56}$$

This relationship allows the development of a design strategy, which we discuss in the next section to assist in the selection of the parameters for a CWS-MAC implementation.

CWS-MAC operation is not contingent on the selection of a specific slot assignment scheme, although the chosen scheme can significantly impact network performance. To accommodate changes in network topology due to node mobility, node failure, and changes in the link quality, TDMA-based medium access schemes designed

for wireless sensor networks typically incorporate a dynamic assignment scheme based on a reservation phase followed by one or more data transmission phases as introduced in [71]. In these approaches, the overhead of the assignment process varies with the frequency of the reservation phase. A detailed discussion of distributed slot assignment approaches is provided in Chapter II.

4. CWS-MAC Parameter Selection

In this subsection, we present a potential strategy for selecting values for the parameters identified earlier. These parameters include the slot size t_s, the minislot size t_{ms}, the number of minislots k, the length of t_b and t_{IFS}. In a given implementation, we begin with the physical characteristics of the network, which include the network data rate, the number of nodes in the network, and the maximum internode spacing from which we can determine the maximum propagation time t_{prop}. As with any TDMA-based scheme, slot synchronization (as discussed in Chapter II) is required and the precision of the time synchronization algorithm utilized determines the guard band t_{guard}. The size of the contention packets is often determined by the complexity of the sensor node control instructions and the size of the acknowledgements can be fixed in the protocol design to be as small as possible. Respectively, these provide t_c and t_{ack}.

The contention beacon length, t_b, and the interframe space, t_{IFS}, should be selected to meet the constraints of (52) and (53). As we shall see when we examine the performance of the contention mode, the probability of successful contention packet transmission in a given contention slot is a function of the number of minislots, k. This, in turn, impacts delay performance in the contention mode. Accordingly, k should be selected to be large enough to achieve a probability of successful transmission that will allow one to meet the desired minimum delay bound. Once we have fixed k, we can now calculate t_{nc} as in (56) to maximize data packet throughput.

Having effectively implemented per flow medium access service, we now turn our attention to the traffic-adaptive mechanism.

B. TRAFFIC ADAPTIVE MECHANISM

To realize the potential performance gains indentified in the previous chapter, we propose a traffic-adaptive mechanism in this section that utilizes flow-specific queue size statistics to allow the per flow mechanism provided by CWS-MAC to respond to changes in flow load and overall network contention. To facilitate the performance analysis of this mechanism, we also develop a general traffic-adaptive, flow-specific medium access control performance model. We examine the two-flow and single-flow cases in detail and demonstrate that contention, non-contention and hybrid schemes are special cases of this general flow-specific model.

1. Traffic-adaptive, Flow-specific Medium Access Mechanism

Assuming each flow (or each set of flows if we choose to group a set of flows with similar characteristics together) has its own queue at each node, we use this queue size as an indicator of flow-specific traffic contention. Queue size has been used extensively, both implicitly and explicitly, as a measure of congestion across a network [41]. As local buffers fill up, strategies include explicit control packet information to "choke" the flow from the sender as well as different packet dropping approaches, such as [149] and its many variants, that lead to retransmissions and implicit congestion notification. The use of queue size has also begun to migrate into wireless sensor network traffic estimation. For example, although TRAMA [27] does not explicitly exchange queue sizes, it does exchange schedules that signal the presence of packets in the local buffers. As an alternate to queue size, network load in the form of contention can be estimated directly by measuring the loss rate associated with acknowledgement packets or indirectly by measuring the channel noise level [28]. The drawback of these approaches to traffic estimation is that they are not flow-specific and therefore do not facilitate flow-specific medium access decision-making. Further details on these alternate approaches to traffic estimation are provided in Chapter II.

The proposed queue-based, traffic-adaptive, flow-specific medium access mechanism operates as follows. As flow load reaches a predetermined threshold, measured in terms of the flow-specific queue size, the flow is switched from one access

mode to another. Each flow (or each set of flows) will have its own queue and associated thresholds. These thresholds, $\theta_{1,f,m}$ and $\theta_{2,f,m}$, define the switching point discussed in the previous chapter and can be unique for each flow *f* and medium access mode *m* as shown in Figure 28. The single-flow, two-mode (contention and non-contention) case is illustrated in Figure 29. When the queue size reaches $\theta_{1,f}$, flow *f* is switched from contention-based to non-contention-based medium access. Similarly, when the queue size drops to $\theta_{2,f}$, the flow is switched from non-contention-based back to contention-based medium access. In the next section, we develop a general model that provides insight into the choice of these thresholds.

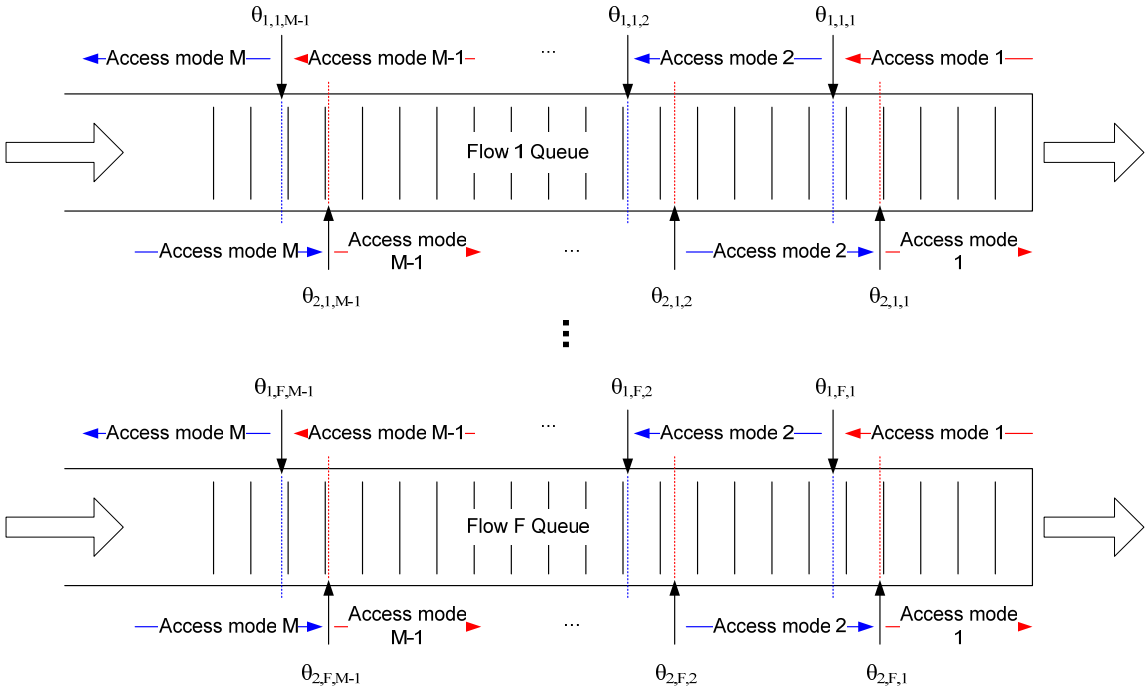


Figure 28. Flow-specific queues and associated thresholds for the general traffic-adaptive, flow-specific medium access model.

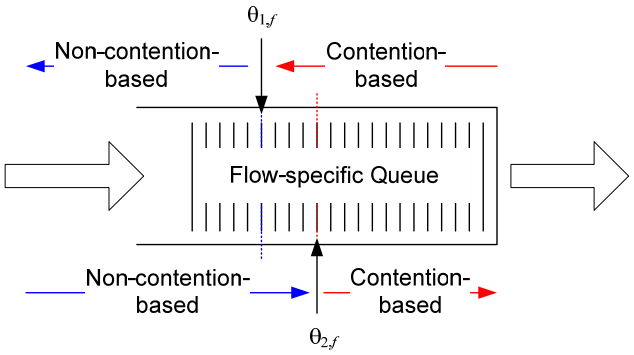


Figure 29. Single-flow, two-mode version of the proposed traffic-adaptive, flow-specific mechanism.

2. General Performance Model for Traffic-adaptive, Flow-specific Medium Access

Traffic-adaptive, flow-specific medium access can be modeled as a finite state machine as shown in Figure 30. Each state is uniquely specified by a vector that reflects the access mode of each flow. The number of states required, Φ, is, therefore, a function of the number of flows, *F*, and the number of unique medium access modes, *M*, as $\Phi = (M)^F$. If we assume that the underlying, individual queues are M/M/1, then this finite-state model can be viewed as a hidden Markov model [150]. To determine the steady state probabilities π_s associated with the individual observable states *s*, we must first derive the state probabilities of the hidden Markov model and then establish the relationships between these Markov states and the observable states. With these steady state probabilities, the mean throughput *S* and delay *D* for the flow-specific medium access scheme can then be developed as

$$S = \sum_{s=1}^{\Phi} \pi_s S_s \text{ and } D = \sum_{s=1}^{\Phi} \pi_s D_s \tag{57}$$

where S_s and D_s are the mean throughput and delay, respectively, experienced in state *s*. In the special case where $\theta_{1,f,m} = \theta_{2,f,m}$ (i.e., a system with no hysteresis), each probability π_s is a function of a unique, non-overlapping set of the underlying Markov state probabilities.

In general, the medium access scheme for flow *f* will transition from one access mode m_i to the next mode m_{i+1} (state *s* to state $s+1$ in Figure 30) to when the number of packets in the flow-specific queue reaches the threshold θ_{1,f,m_i} (denoted by $\alpha_{s,s+1}$ in Figure 30). Similarly, the transition from state $s+1$ to state *s* occurs when the number of packets drops to $\theta_{2,f,m}$ (denoted by $\beta_{s+1,s}$). The probability of these transitions is a function of both the number of packets N_f in the flow-specific queue *f* and the utilization in the current observed state. The utilization $\rho_{s,f}$ is defined as the ratio of the packet arrival rate for flow *f* to the service time for flow *f* and is unique to the state *s* and the flow *f*. Given the result for a M/M/1 queue that the total number of customers *N* in the system is [48]

$$N = \frac{\rho}{1-\rho}, \tag{58}$$

the utilization can be derived from Little's Law [48] as

$$\rho_{s,f} = \frac{\lambda_f D_s}{1+\lambda_f D_s} \tag{59}$$

where λ_f is the packet arrival rate for flow *f* and D_s is the mean delay in state *s*. In the following section, we examine this relationship closer for the two-flow, two-mode case and develop both throughput and delay expressions for the example of Chapter III.

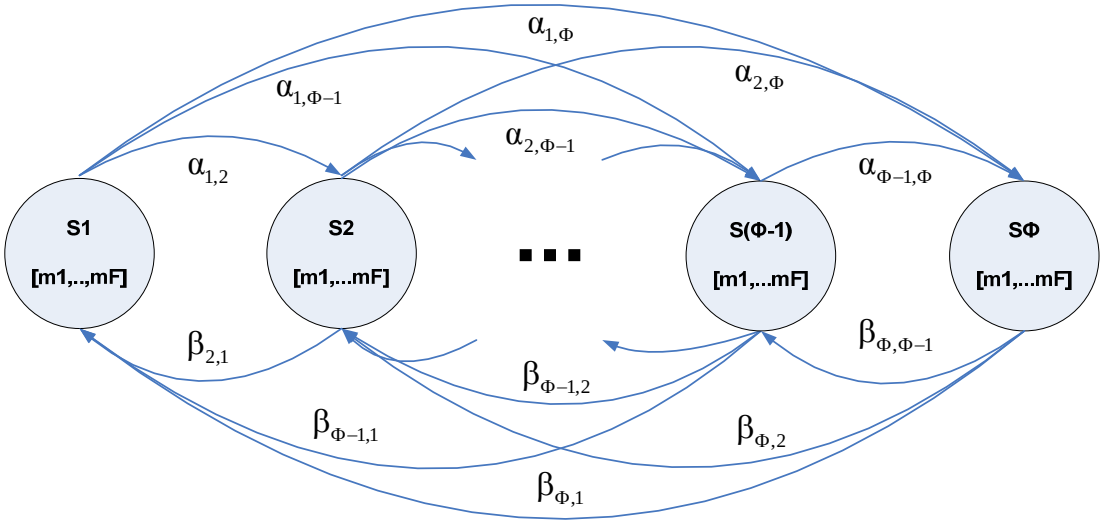


Figure 30. General traffic-adaptive, flow-specific finite state model.

3. Two-flow, Two-mode Case: Flow-specific Medium Access

As shown in Figure 31, it requires a four-state model to represent a two-flow, flow-specific medium access scheme such as one capable of providing both contention and non-contention access modes. We can make a set of simplifying assumptions to allow us to compare the performance of this traffic-adaptive mechanism to that of the ideal case in the example of Chapter III. Without a loss of generality, we assume that it is flow 1 that has a constant arrival rate (analogous to the control flow in the example of Chapter III) and remains in the contention-based access mode while flow 2 is allowed to transition between access modes as its arrival rate varies (analogous to the data flow in the example). Accordingly, $\alpha_{1,2}$ and $\beta_{2,1}$ are the only non-zero transition rates since states S3 and S4 are not achievable and the full model of Figure 31 can be reduced to the two-state model of Figure 32.

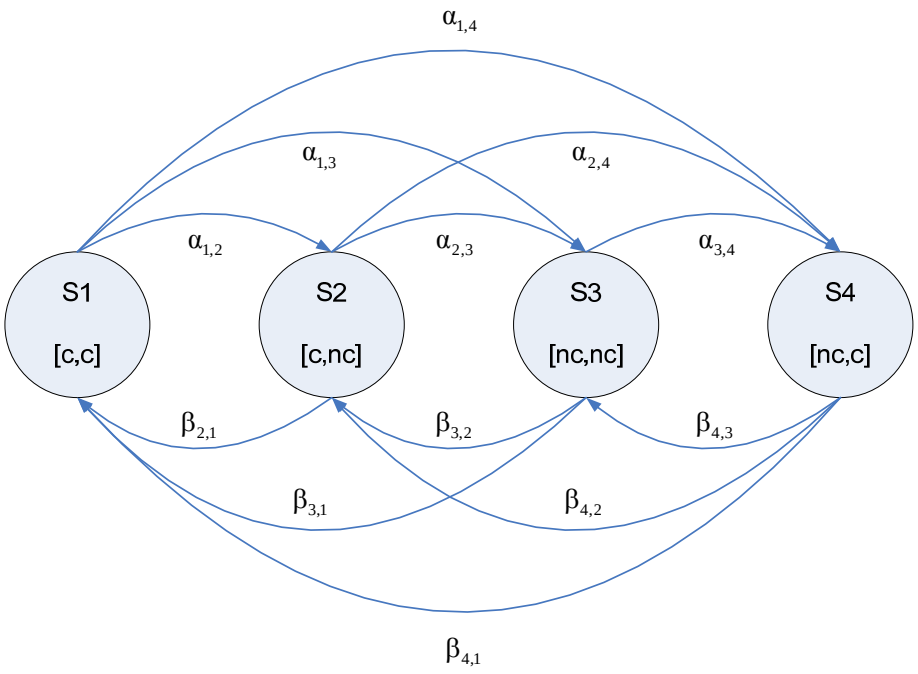


Figure 31. Full 4-state model for two-flow flow-specific, medium access.

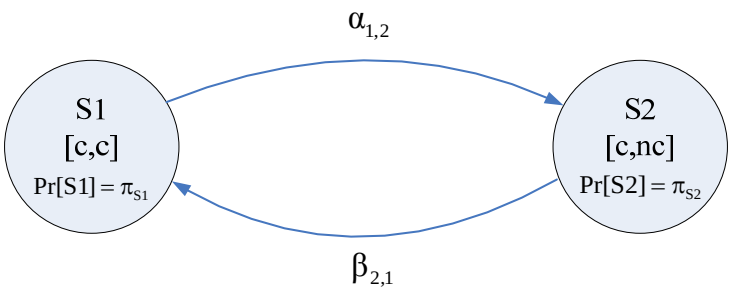


Figure 32. Simplified 2-state model for two-flow flow-specific, medium access.

Assuming that the underlying Markov process is M/M/1, the bilevel hysteretic service rate control work of [151] can be adopted to arrive at the steady state probabilities by viewing the system as having two distinct service rates μ_1 and μ_2 (corresponding to the states S1 and S2). The states of this underlying Markov Chain are defined by the state the system is in (S1 or S2 from Figure 32) and the queue size (number of packets awaiting transmission). The transition from μ_1 to μ_2 occurs when the number of packets in the queue of flow 2 reaches $\theta_{1,2}$ and the transition from μ_2 to μ_1 occurs when the number of packets in the queue of flow 2 drops to $\theta_{2,2}$, as shown in Figure 29. Examining the underlying Markov model, shown in Figure 33, the state probabilities P_n (where n is the queue length) are given by [151]

$$
\begin{aligned}
P_n &= \left(\rho_{1,2}\right)^n P_0 && n \subset \left[0, \theta_{2,2}-1\right] \\
P_{\theta_{2,2}+i} &= \frac{\left(\rho_{1,2}\right)^{\theta_{1,2}-1} \rho_{2,2}\left(1-\rho_{1,2}\right)\left(\left(\rho_{1,2}\right)^i - \left(\rho_{1,2}\right)^\Delta\right)\left(1-\left(\rho_{2,2}\right)^i\right)}{\left(1-\left(\rho_{1,2}\right)^\Delta\right)\left(1-\rho_{1,2}\right)\left(1-\rho_{2,2}\right)} P_0 && i \subset \left[0, \Delta-1\right] \qquad (60) \\
P_{\theta_{1,2}+j-1} &= \frac{\left(\rho_{1,2}\right)^{\theta_{1,2}-1}\left(1-\rho_{1,2}\right)\left(1-\left(\rho_{2,2}\right)^\Delta\right)}{\left(1-\left(\rho_{1,2}\right)^\Delta\right)\left(1-\rho_{2,2}\right)}\left(\rho_{2,2}\right)^j P_0 && j \subset \left[1, \infty\right]
\end{aligned}
$$

where

$$
P_0 = \left(\frac{1}{\left(1-\rho_{1,2}\right)} - \frac{\Delta\left(\rho_{1,2}\right)^{\theta_{1,2}-1}\left(\rho_{1,2}-\rho_{2,2}\right)}{\left(1-\left(\rho_{1,2}\right)^\Delta\right)\left(1-\rho_{2,2}\right)} \right)^{-1} \qquad (61)
$$

and $\Delta = \theta_{1,2} - \theta_{2,2}$ captures the extent of the hysteresis loop created by $\theta_{1,2}$ and $\theta_{2,2}$.

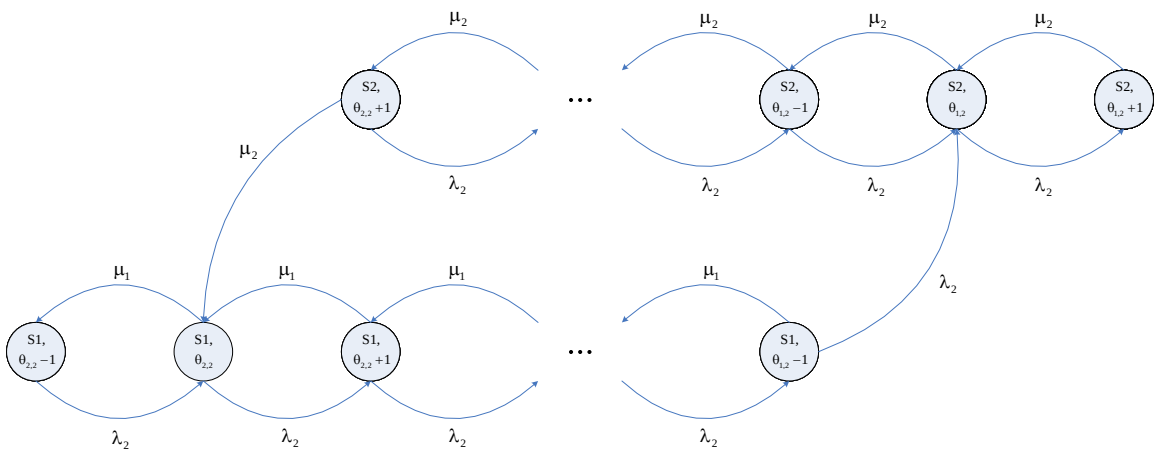


Figure 33. Underlying Markov Chain for two-flow, two-mode example.

Returning to our two-state medium access model of Figure 32, the probability that the system is in state S2, π_{S2}, is equivalent to the probability that the system is in service rate μ_2, which can be shown to be [151]

$$\pi_{S2} = \frac{\Delta\left(\rho_{1,2}\right)^{\theta_{1,2}-1}\left(\rho_{2,2}\right)\left(1-\rho_{1,2}\right)}{\left(1-\left(\rho_{1,2}\right)^{\Delta}\right)\left(1-\rho_{2,2}\right)} P_0 . \tag{62}$$

The probability that the system is in state S1 is then simply $\pi_{S1} = 1 - \pi_{S2}$. The steady state probabilities are plotted as a function of $\theta_{1,2}$ in Figure 34. It can be seen that, as expected, for the limiting cases of $\theta_{1,2}$ approaching zero and $\theta_{1,2}$ approaching infinity, the probability that the flow-specific medium access scheme is in State 2 approaches one ($\pi_{S2} = 1$) and zero ($\pi_{S2} = 0$), respectively.

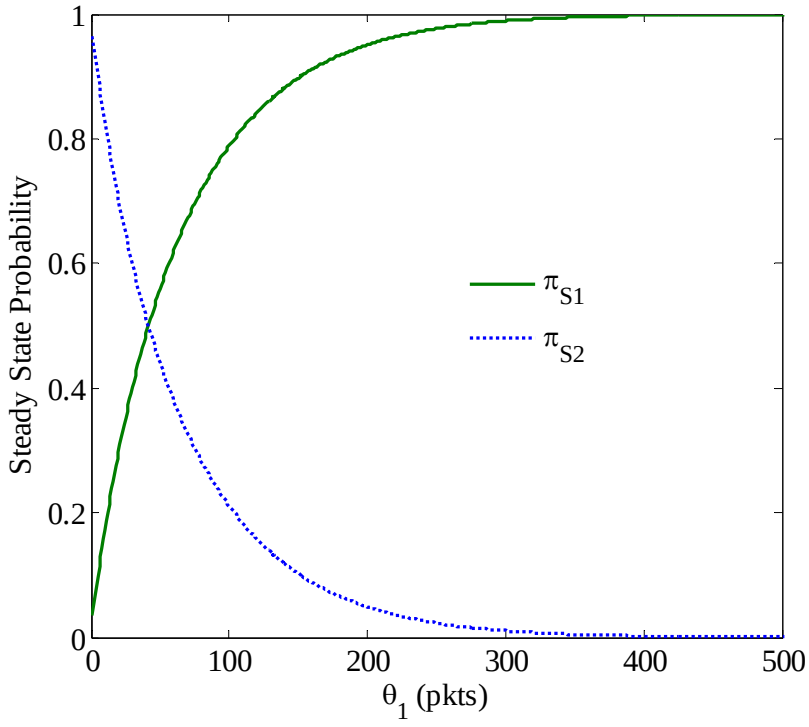


Figure 34. Steady state probability for two-flow, two-mode model as a function of the queue-based threshold, θ_1.

Following the analysis of Chapter III, the associated state throughputs and delays are

$$\begin{aligned} S_{S1} &= S^c \text{ and } S_{S2} = \frac{\lambda_1}{\lambda} S_1^c + \frac{\lambda_2}{\lambda} S_2^{nc} \\ D_{S1} &= D^c \text{ and } D_{S2} = \frac{\lambda_1}{\lambda} D_1^c + \frac{\lambda_2}{\lambda} D_2^{nc} \end{aligned} \tag{63}$$

where S^c and D^c are the throughput and delay, respectively, of the aggregate flow in the contention mode, S_f^c and D_f^c are the throughput and delay, respectively, of flow *f* in the contention mode, and S_f^{nc} and D_f^{nc} are the throughput and delay, respectively, of flow *f* in the non-contention mode. Substituting (62) and (63) into (57), we can then develop the resulting aggregate mean throughput and delay expressions as

$$S = S^c + \frac{\Delta\left(\rho_{1,2}\right)^{\theta_{1,2}-1}\left(\rho_{2,2}\right)\left(1-\rho_{1,2}\right)}{\left(1-\left(\rho_{1,2}\right)^{\Delta}\right)\left(1-\rho_{2,2}\right)} P_0\left(\frac{\lambda_1}{\lambda} S_1^c + \frac{\lambda_2}{\lambda} S_2^{nc} - S^c\right)$$
$$D = D^c + \frac{\Delta\left(\rho_{1,2}\right)^{\theta_{1,2}-1}\left(\rho_{2,2}\right)\left(1-\rho_{1,2}\right)}{\left(1-\left(\rho_{1,2}\right)^{\Delta}\right)\left(1-\rho_{2,2}\right)} P_0\left(\frac{\lambda_1}{\lambda} D_1^c + \frac{\lambda_2}{\lambda} D_2^{nc} - D^c\right). \quad (64)$$

This analysis of the special case where the demand of one flow is fixed can be extended to the two-flow, *M*-mode case with $M > 2$ using the more general variable service rate work of [152]. It should be noted that the use of the work in [151] (or, alternately, [152]) in this analysis to arrive at the steady state probabilities is not without its limitations. We will discuss these in detail when we examine the simulation results at the end of this chapter.

Using the parameters of the example in Chapter III, we plot mean aggregate delay as a function of normalized load in Figure 35 for $\theta_1 = 20$ and $\theta_2 = 5$. It can be seen that, as expected, the flow-specific scheme performs as well as CSMA when the aggregate load is low and outperforms all three approaches when a flow exists in both the contention and non-contention modes. The role of θ_1 as the switching point can be clearly seen in Figure 36 where we plot both delay and throughput as a function of load for various values of θ_1. At the optimum value for θ_1 (close to 20 packets in this example), the mechanism transitions to contention-free operation as the delay curves intersect. At values below optimum, the scheme transitions too early and the aggregate delay at low loads suffers. For values of θ_1 above optimum, the scheme transitions late and the heavy load begins to overwhelm the contention-based mode, the delay grows and the throughput saturates (and will eventually drop off).

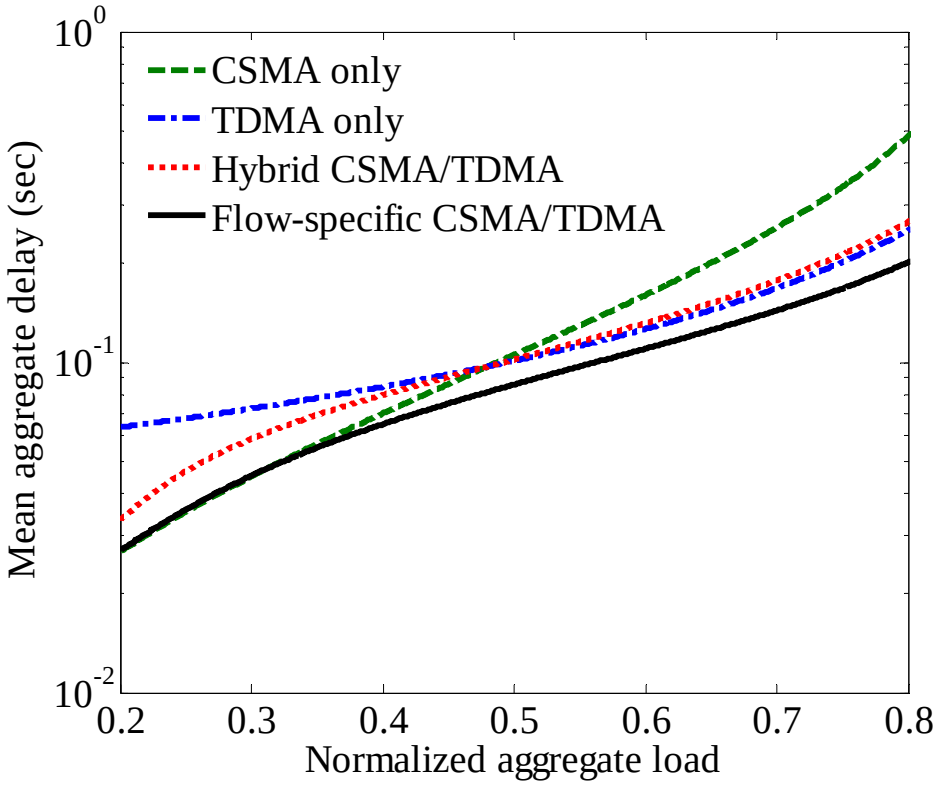
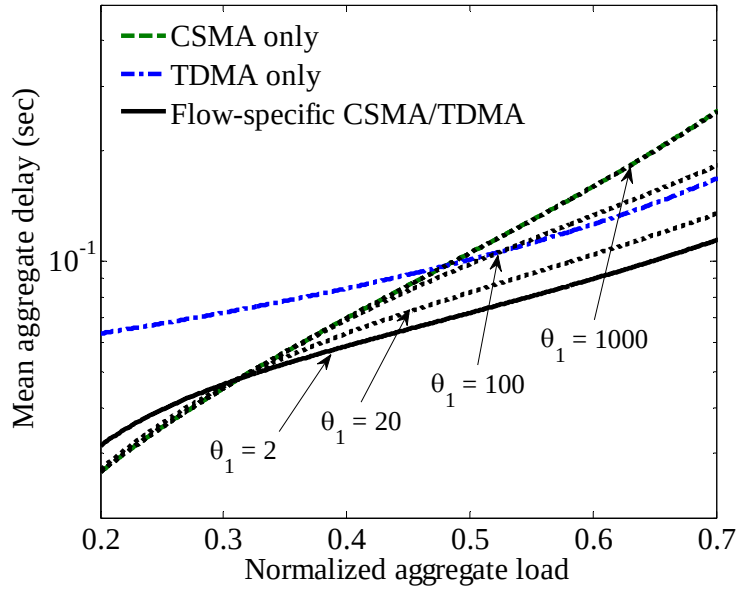
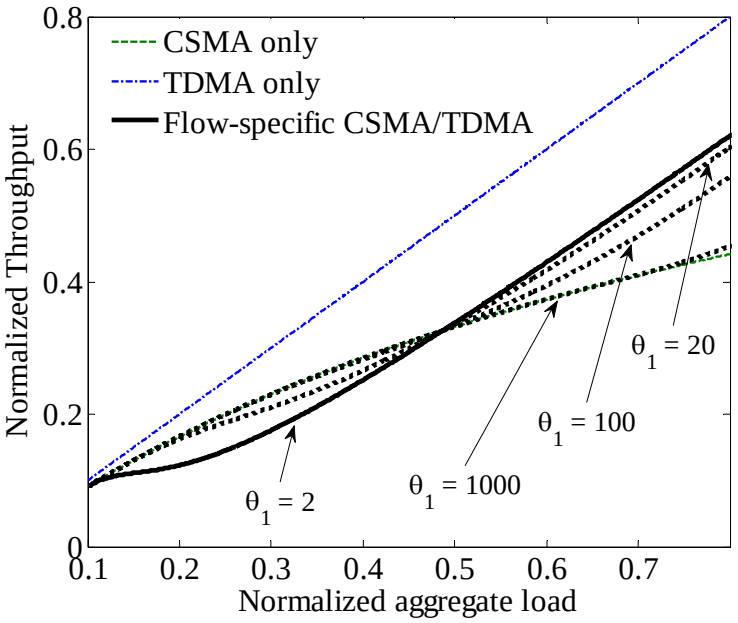


Figure 35. Packet delay plotted as a function of normalized load for slotted nonpersistent CSMA, TDMA, hybrid using CSMA/TDMA and flow-specific medium access using CSMA/TDMA with $\theta_1 = 20$ and $\theta_2 = 5$. Channel rate is 1 Mbps, packet size is 1000 bits, there are 100 slots in a TDMA frame (each slot is one packet length in duration) and a = 0.01 for the CSMA schemes. The CSMA plot assumes steady state and represents minimum achievable delay.



(a)



(b)

Figure 36. Mean aggregate (a) delay and (b) throughput plotted as a function of the normalized aggregate load for multiple values of θ_1. Channel rate is 1 Mbps, packet size is 1000 bits, there are 100 slots in a TDMA frame (each slot is one packet length in duration) and a = 0.01 for the CSMA schemes. The CSMA plot assumes steady state and represents minimum achievable delay.

4. Single-flow, Two-mode Case: Hybrid Medium Access

The model of Figure 32 can be further simplified if we examine the single-flow case. This case can be shown to be equivalent to the hybrid case where multiple flows are treated in aggregation. Thus, hybrid approaches represent a special case of the more general flow-specific approach. To demonstrate this, we note that there is a single, aggregate queue in a hybrid scheme, so $\theta_{1,2}$ and $\theta_{2,2}$ reduce to θ_1 and θ_2, respectively, and $\rho_{1,2}$ and $\rho_{2,2}$ reduce to $\rho_1 = \rho_c$ and $\rho_2 = \rho_{nc}$, respectively. Following the analysis of the previous section, the state probabilities are

$$\begin{aligned} \pi_{nc} &= \frac{\Delta(\rho_c)^{\theta_1 - 1}(\rho_{nc})(1-\rho_c)}{\left(1-(\rho_c)^{\Delta}\right)(1-\rho_{nc})} P_0 \\ \pi_c &= 1 - \pi_{nc} \end{aligned} \tag{65}$$

where

$$P_0 = \left(\frac{1}{(1-\rho_c)} - \frac{\Delta(\rho_c)^{\theta_1 - 1}(\rho_c - \rho_{nc})}{\left(1-(\rho_c)^{\Delta}\right)(1-\rho_{nc})} \right)^{-1} \tag{66}$$

and $\Delta = \theta_1 - \theta_2$.

Since S_1 and S_2 are equivalent to S^c and S^{nc}, respectively, and D_1 and D_2 are equivalent to D^c and D^{nc}, respectively, the overall mean throughput is $S = \pi_c S^c + \pi_{nc} S^{nc}$ and the mean delay is $D = \pi_c D^c + \pi_{nc} D^{nc}$. Substituting (65) into these expressions, we arrive at

$$\begin{aligned} S &= S^c + \frac{\Delta(\rho_c)^{\theta_1 - 1}(\rho_{nc})(1-\rho_c)}{\left(1-(\rho_c)^{\Delta}\right)(1-\rho_{nc})} P_0 \left(S^{nc} - S^c\right) \\ D &= D^c + \frac{\Delta(\rho_c)^{\theta_1 - 1}(\rho_{nc})(1-\rho_c)}{\left(1-(\rho_c)^{\Delta}\right)(1-\rho_{nc})} P_0 \left(D^{nc} - D^c\right). \end{aligned} \tag{67}$$

This result is included in Figure 35 for $\theta_1 = 20$ and $\theta_2 = 5$.

5. Single-flow, Single-mode Case: Contention and Contention-free Access

Finally, it is straightforward to show that the contention only [20],[21] and non-contention [26] schemes are trivial single-flow, single-mode cases of the general flow-specific model. The state probabilities for the contention-based schemes are $\pi_{S1} = \pi_c = 1$ and $\pi_{S2} = \pi_{nc} = 0$ while they are $\pi_{S1} = \pi_c = 0$ and $\pi_{S2} = \pi_{nc} = 1$ for the non-contention-based scheme. Substituting these into (57), we arrive $S = S^c$ and $D = D^c$ for the contention-based scheme and $S = S^{nc}$ and $D = D^{nc}$ for the non-contention-based scheme, as expected.

C. TRAFFIC-ADAPTIVE CWS-MAC

We now bring together the flow-specific medium access service of CWS-MAC and the queue-based, traffic-adaptive mechanism described in the previous section. The flow-specific queues are maintained at the link layer, which implies a cross-layer solution in which the link layer mechanism is capable of determining which flow a packet is associated with. The queue size measurement is taken whenever a packet is added or removed from the applicable queue. It should be noted that while a non-contention mode packet is removed from the queue upon transmission, a contention mode packet is not removed until the appropriate acknowledgement is received at the sender. Although certainly not required, this queue size measurement can be smoothed by applying a moving average to it. When a flow is transitioned from one mode to another, all queued packets within that flow are transitioned as well. This has the effect of "clearing" out the flow from the prior access mode and specifically improves delay and throughput recovery time when a flow is transitioned from the contention mode to the non-contention mode.

The distributed nature of traffic-adaptive CWS-MAC leads to a subtle point that should not be overlooked: neighboring nodes may assign the same flow to different medium access modes. This is because the state transitions of Figure 31 are based on local queue statistics, which will vary between neighboring nodes. This does not pose a problem in traffic-adaptive CWS-MAC because the medium access mode is specific to the sender not the receiver and a receiver needs no prior arrangement to receive a flow in

a given mode. Hence, although a node may receive a flow in one mode, it requires no coordination to switch to reception in the other mode and it is free to retransmit the flow in either mode.

D. PERFORMANCE ANALYSIS OF TRAFFIC-ADAPTIVE CWS-MAC

In this section, we develop individual expressions for the non-contention mode and contention mode throughput and delay for traffic-adaptive CWS-MAC. We then combine these using (64) to arrive at the overall mean delay and throughput for the example of the previous chapter. In conjunction with this analysis, we develop the throughput and delay performance of a slotted ALOHA queuing system in which the server takes periodic vacations during which times it is unavailable to serve the queues.

1. Non-contention Throughput for CWS-MAC

We begin with the non-contention throughput. At steady state, the arrival rate is equivalent to the departure rate and the normalized non-contention throughput, S^{nc}, is given by [153]

$$S^{nc} = \frac{\Lambda_{nc}\bar{L}_{nc}}{R} \tag{68}$$

where Λ_{nc} is the mean total arrival rate for the non-contention mode, $\bar{L}_{nc}$ is the mean packet size (in bits) for the non-contention mode, and *R* is the channel rate in bps. For a TDMA-based scheme, this throughput is bounded by a maximum value that is dependent on the per frame overhead. Specifically,

$$S^{nc}_{\max} = \frac{M\bar{T}_{data}}{t_f} \tag{69}$$

where $\bar{T}_{data}$ is the mean time spent in a slot transmitting useful data, t_f is the frame length in seconds and we have assumed, without loss of generality, that each node is assigned a single slot in the frame. Clearly, $M\bar{T}_{data} = t_f - \bar{T}_{ovrhd}$ where $\bar{T}_{ovrhd}$ is the mean time spent in a frame on overhead and (69) can alternately be written as

$$S^{nc}_{\max} = 1 - \frac{\bar{T}_{ovrhd}}{t_f}. \tag{70}$$

To calculate $\bar{T}_{data}$, we must account for both the non-contention and contention slots in Figure 27. In the first case, T_{data} for a packet transmitted in a non-contention slot is equal to t_{nc}. This value is reduced in the case of the contention slot by the overhead associated with the contention access mode, which can be seen in Figure 27 to be $t_b + kt_{ms}$. Combining these cases (and accounting for the t_{IFS} term), we have

$$T_{data} = \begin{cases} t_{nc} & \text{for a non-contention slot} \\ t_{nc} + t_{IFS} - t_b - kt_{ms} & \text{for a contention slot} \end{cases} \tag{71}$$

for *k* minislots per slot and

$$t_{nc} = t_s - t_{IFS} - t_{prop} - t_{guard}\,. \tag{72}$$

We can calculate $\bar{T}_{data}$ then as

$$\begin{aligned} \bar{T}_{data} = {} & T_{data}[\text{non-contention slot}] \times \Pr[\text{non-contention slot}] \\ & + T_{data}[\text{contention slot}] \times \Pr[\text{contention slot}], \end{aligned} \tag{73}$$

which, from (71), can be written as

$$\begin{aligned} \bar{T}_{data} = {} & t_{nc} \Pr[\text{non-contention slot}] \\ & + \left(t_{nc} + t_{IFS} - t_b - kt_{ms}\right) \Pr[\text{contention slot}]. \end{aligned} \tag{74}$$

Assuming that the contention mode arrivals follow a Poisson distribution, the probabilities in (74) can be derived as follows. The probability that a slot is designated as a non-contention slot is equivalent to the probability that there are no control packet arrivals during the previous slot and that no residual control packet retransmissions are pending from the previous contention slot. For now, we will assume that the probability of the latter is negligible (we will come back to this point in a follow-on section). Focusing then on the first term,

$$\Pr[\text{non-contention slot}] = \Pr[\text{no contention packets arrivals in previous slot}]. \tag{75}$$

Since the contention packet arrivals are Poisson, this can be shown to be

$$\Pr[\text{non-contention slot}] \equiv p_0 = e^{-\Lambda_c t_s} \tag{76}$$

where the aggregate control mode packet arrival rate $\Lambda_c = M\lambda_c$ for *M* nodes. The Pr[contention slot] is simply $1 - \Pr[\text{non-contention slot}]$ or

$$\Pr[\text{contention slot}] = 1 - p_0 = 1 - e^{-\Lambda_c t_s}\,. \tag{77}$$

Substituting (76) and (77) into (74), we have

$$\overline{T}_{data} = t_{nc} e^{-\Lambda_c t_s} + \left(t_{nc} + t_{IFS} - t_b - k t_{ms}\right)\left(1 - e^{-\Lambda_c t_s}\right). \tag{78}$$

Rearranging terms,

$$\overline{T}_{data} = t_{nc} + \left(1 - e^{-\Lambda_c t_s}\right)\left(t_{IFS} - t_b - k t_{ms}\right), \tag{79}$$

and substituting into (69),

$$S_{\max}^{nc} = \frac{M\left(t_{nc} + \left(1 - e^{-\Lambda_c t_s}\right)\left(t_{IFS} - t_b - k t_{ms}\right)\right)}{t_f}. \tag{80}$$

Recognizing that $t_s = \frac{t_f}{M}$, we finally arrive at

$$S_{\max}^{nc} = \frac{t_{nc} + \left(1 - e^{-\Lambda_c t_s}\right)\left(t_{IFS} - t_b - k t_{ms}\right)}{t_s}. \tag{81}$$

Combining (81) and (68), we can express the throughput for the non-contention mode as

$$S^{nc} = \begin{cases} \dfrac{\Lambda_{nc}\overline{L}}{R} & \Lambda_{nc} \le S_{\max}^{nc}\left(\dfrac{R}{\overline{L}}\right) \\ S_{\max}^{nc} & \text{otherwise} \end{cases}. \tag{82}$$

To capture the effect of the ratio of T_{data} for the contention slot to that for a non-contention slot or, equivalently, the percentage of the bandwidth allocated to the contention flow, we define β as

$$\beta = \frac{t_b + k t_{ms}}{t_s}. \tag{83}$$

Thus, $\beta \in [0,1]$ and, if the timing parameters in (83) are fixed, is proportional to the choice of k. A larger value of β represents a larger percentage of bandwidth allocated to the contention mode. A plot of maximum non-contention throughput as a function of the probability of a non-contention slot p_0 for various values of β is provided in Figure 37.

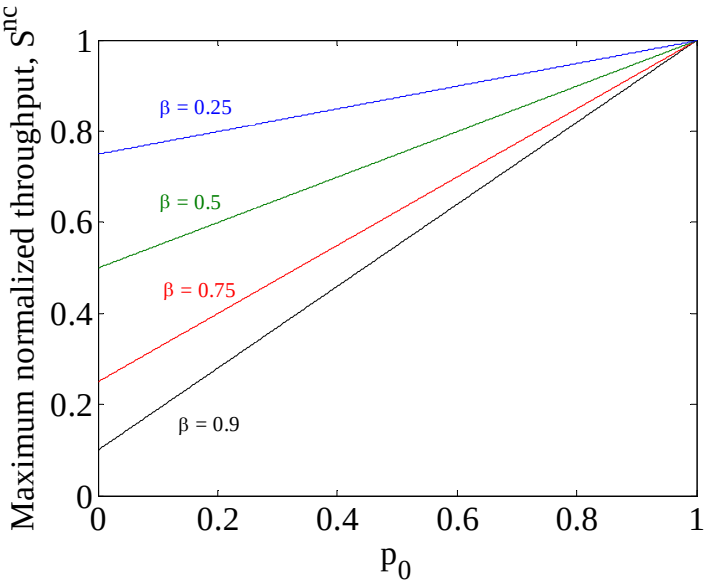


Figure 37. Maximum non-contention throughput as a function of a function of the probability of a non-contention slot p_0 for various values of β.

2. Non-contention Mean Delay for CWS-MAC

Turning our attention to latency, the mean delay of a packet in the non-contention mode is comprised of four parts [153]: (1) $\overline{T}_{sync}^{nc}$, the mean delay associated with waiting for the next slot boundary (sometimes referred to as the synchronization delay); (2) $\overline{T}_{w}^{nc}$, the mean waiting time in the queue, (3) $\overline{T}_{xmt}^{nc}$, the mean transmission time, and (4) t_{prop}, the maximum propagation time of the packet. This is summarized as

$$\overline{D}^{nc} = \overline{T}_{sync}^{nc} + \overline{T}_{w}^{nc} + \overline{T}_{xmt}^{nc} + t_{prop}. \tag{84}$$

To develop expressions for the first three terms, we must examine the two cases identified in the previous section. The first term, $\overline{T}_{sync}^{nc}$, is the same in both cases. Since the non-contention packet arrivals are assumed to be purely random (Poisson distributed) and the period of arrival is the slot duration t_s, then the synchronization delay is simply

$$\overline{T}_{sync}^{nc} = \frac{t_s}{2}. \tag{85}$$

To calculate the mean waiting time in the queue, $\overline{T}_{w}^{nc}$, we again assume Poisson arrivals and can therefore view the network as a set of identical M/G/1 queues where the

mean arrival rate is λ_{nc}. To develop the effective service time distribution, we begin by calculating the effective service time $T_{s_1}^{nc}$ for a packet that is transmitted in a single non-contention slot. Without a loss of generality, we will assume that exactly one non-contention packet is transmitted during a non-contention slot and that each node (i.e., queue) is assigned a single slot in each frame. Thus, in this case, each queue services one packet in a frame and the effective service time is simply the frame time as in

$$T_{s_1}^{nc} = t_f \,. \tag{86}$$

Note that since the propagation time is specifically included in our slot time calculations (and, hence, our frame time calculations), we have implicitly included it in our effective service time.

If the first slot encountered by a packet is a contention slot, then the effective service time of a packet is increased because, as we saw in the previous section, T_{data} for a contention slot is reduced relative to that for a non-contention slot. Accordingly, the packet will be serviced over multiple slots or, equivalently, multiple frames. Let us define κ as the smallest integer, greater than or equal to the ratio of T_{data} for the non-contention slot to that for a contention slot. From (71),

$$\kappa = \left\lceil \frac{t_{nc}}{t_{nc} + t_{IFS} - t_b - kt_{ms}} \right\rceil \tag{87}$$

where $f(x) = \lceil x \rceil$ is the ceiling operator. Now, let us examine the case where a packet is serviced by a contention slot followed by a non-contention slot. In this case, the effective service time will simply be twice the frame time, t_f. Given our assumption that exactly one non-contention packet is transmitted during a non-contention slot, with some thought it can be seen that, in general, a packet service time will terminate when either the packet experiences a non-contention slot or it has spanned across κ contention slots. The service time is therefore a discrete random variable that can take on the values $\left[t_f, 2t_f, ..., (\kappa-1)t_f, \kappa t_f\right]$. By use of the ceiling operator in (87), we have made the conservative assumption that when a packet transmission spans multiple slots, any residual slot time in the last slot remains unfilled. Accordingly, our service time

expression can be viewed as an upper bound, which can be improved upon by allowing subsequent packets to make use this residual slot time.

We can derive the probability distribution for T_s^{nc} by examining the individual cases. The probability that a packet will encounter a single non-contention slot is simply the probability that the first slot it encounters is a non-contention slot, which, from (76), is $p_0 = e^{-\Lambda_c t_s}$. The probability that T_s^{nc} will span exactly two frames is the probability of a contention slot followed by a non-contention slot or $p_0(1-p_0)$. Extending this through the case of $(\kappa-2)$ contention slots followed by a non-contention slot, we have

$$\Pr\left[T_s^{nc} = \alpha t_f\right] = p_0(1-p_0)^{\alpha-1} \qquad \text{for } 1 \le \alpha < \kappa \tag{88}$$

where α is an integer. The probability for the terminating case in which we have either $(\kappa-1)$ contention slots followed by a non-contention slot or κ consecutive contention slots is then

$$\Pr\left[T_s^{nc} = \kappa t_f\right] = 1 - \sum_{i=1}^{\kappa-1} p_0(1-p_0)^{i-1}. \tag{89}$$

Substituting $j = i-1$ and rearranging,

$$\Pr\left[T_s^{nc} = \kappa t_f\right] = 1 - p_0 \sum_{j=0}^{\kappa-2} (1-p_0)^j. \tag{90}$$

We can now use the well-known identity

$$\sum_{i=0}^{n} c^i = \frac{c^{n+1}-1}{c-1} \tag{91}$$

to simplify (90) to

$$\Pr\left[T_s^{nc} = \kappa t_f\right] = (1-p_0)^{\kappa-1}. \tag{92}$$

Combining (88) and (92) and accounting for the zero probability case of $\alpha > \kappa$, we arrive at the distribution of T_s^{nc} as

$$\Pr\left[T_s^{nc} = \alpha t_f\right] = \begin{cases} p_0(1-p_0)^{\alpha-1} & \text{for } 1 \le \alpha < \kappa \\ (1-p_0)^{\kappa-1} & \text{for } \alpha = \kappa \\ 0 & \text{for } \alpha > \kappa \end{cases}. \tag{93}$$

This result is logical when we observe that T_s^{nc} has the form of a modified geometric random variable. By this, we mean that we count the consecutive number of unsuccessful trials (contention slots, in our case) until the first successful trial (non-contention slot), but we are bounded by a maximum number of trials (κ). The probability and cumulative distribution functions for T_s^{nc} are shown in Figures 38 and 39, respectively.

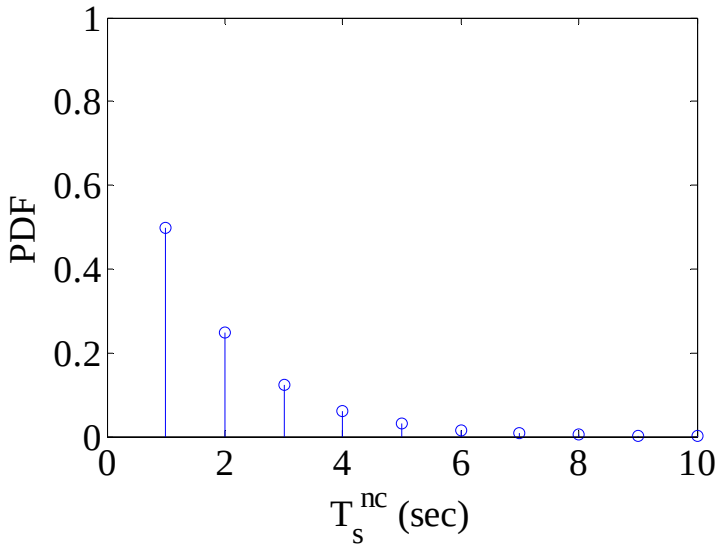


Figure 38. Probability distribution function of T_s^{nc} with $t_f = 1$ sec and $\kappa = 10$.

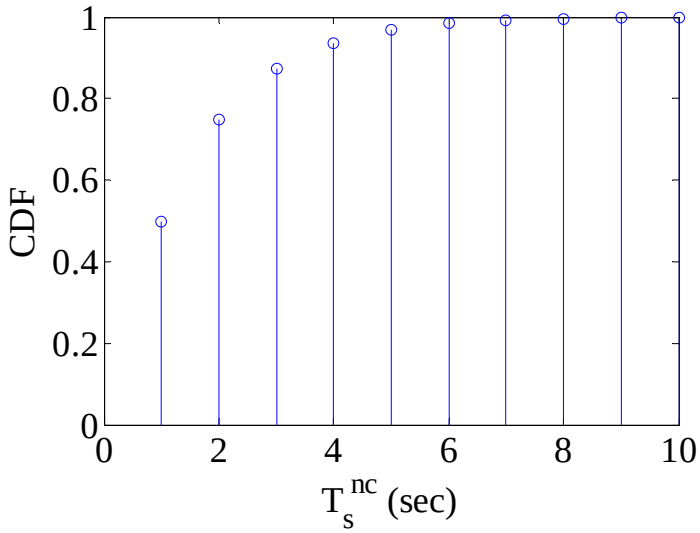


Figure 39. Cumulative distribution function of T_s^{nc} with $t_f = 1$ sec and $\kappa = 10$.

Given the distribution of T_s^{nc}, we can now calculate its mean and variance. The mean is defined as

$$E\left[T_s^{nc}\right] \equiv \bar{T}_s^{nc} = \sum_{i=1}^{\infty}\left(T_{s_i}^{nc} \times \Pr[T_s^{nc} = T_{s_i}^{nc}]\right). \tag{94}$$

Making the appropriate substitutions from (93),

$$\bar{T}_s^{nc} = \sum_{i=1}^{\kappa-1} i t_f\, p_0 \left(1-p_0\right)^{i-1} + \kappa t_f \left(1-p_0\right)^{\kappa-1}. \tag{95}$$

Rearranging and including the case of $i = 0$ in the summation,

$$\bar{T}_s^{nc} = t_f\, p_0 \sum_{i=0}^{\kappa-1} i \left(1-p_0\right)^{i-1} + \kappa t_f \left(1-p_0\right)^{\kappa-1}. \tag{96}$$

To evaluate the summation in the first term, we take the partial derivative of (91)

$$\frac{\delta}{\delta c}\left(\sum_{i=0}^{n} c^i = \frac{c^{n+1}-1}{c-1}\right), \tag{97}$$

which, using the linearity property of the derivative operation, is equivalent to

$$\sum_{i=0}^{n}\left(\frac{\delta}{\delta c} c^i\right) = \frac{\delta}{\delta c}\left(\frac{c^{n+1}-1}{c-1}\right). \tag{98}$$

Evaluating this,

$$\sum_{i=0}^{n} i c^{i-1} = \frac{(n+1)c^n(c-1) - \left(c^{n+1}-1\right)}{(c-1)^2} \tag{99}$$

and, making the substitutions $n = \kappa - 1$ and $c = 1 - p_0$,

$$\sum_{i=0}^{\kappa-1} i \left(1-p_0\right)^{i-1} = \frac{(\kappa-1+1)\left(1-p_0\right)^{\kappa-1}\left(\left(1-p_0\right)-1\right) - \left(\left(1-p_0\right)^{\kappa-1+1}-1\right)}{\left(\left(1-p_0\right)-1\right)^2}. \tag{100}$$

Simplifying, we have

$$\sum_{i=0}^{\kappa-1} i \left(1-p_0\right)^{i-1} = \frac{1-\left(1-p_0\right)^{\kappa} - \kappa p_0 \left(1-p_0\right)^{\kappa-1}}{{p_0}^2}. \tag{101}$$

Substituting (101) into (96),

$$\bar{T}_s^{nc} = t_f\, p_0 \left(\frac{1-\left(1-p_0\right)^{\kappa} - \kappa p_0 \left(1-p_0\right)^{\kappa-1}}{{p_0}^2}\right) + \kappa t_f \left(1-p_0\right)^{\kappa-1}. \tag{102}$$

Rearranging,

$$\bar{T}_s^{nc} = t_f \left(\frac{1-(1-p_0)^{\kappa} - \kappa p_0 (1-p_0)^{\kappa-1} + \kappa p_0 (1-p_0)^{\kappa-1}}{p_0} \right). \tag{103}$$

Canceling like terms, we finally arrive at the mean of T_s^{nc} as

$$\bar{T}_s^{nc} = t_f \left(\frac{1-(1-p_0)^{\kappa}}{p_0} \right) \tag{104}$$

and the square of the mean as

$$\left(\bar{T}_s^{nc}\right)^2 = \left(t_f\right)^2 \left(\frac{1-(1-p_0)^{\kappa}}{p_0} \right)^2. \tag{105}$$

Checking the limiting cases of $p_0 = 1$ (all non-contention slots) and $p_0 = 0$ (all contention slots), we find that, as expected, the mean of T_s^{nc} is t_f in the former and κt_f in the latter. In the $p_0 = 0$ case, this result is arrived at through a single application of L'Hôpital's Rule. A plot of the mean of T_s^{nc} as a function of p_0 for various values of κ is provided in Figure 40.

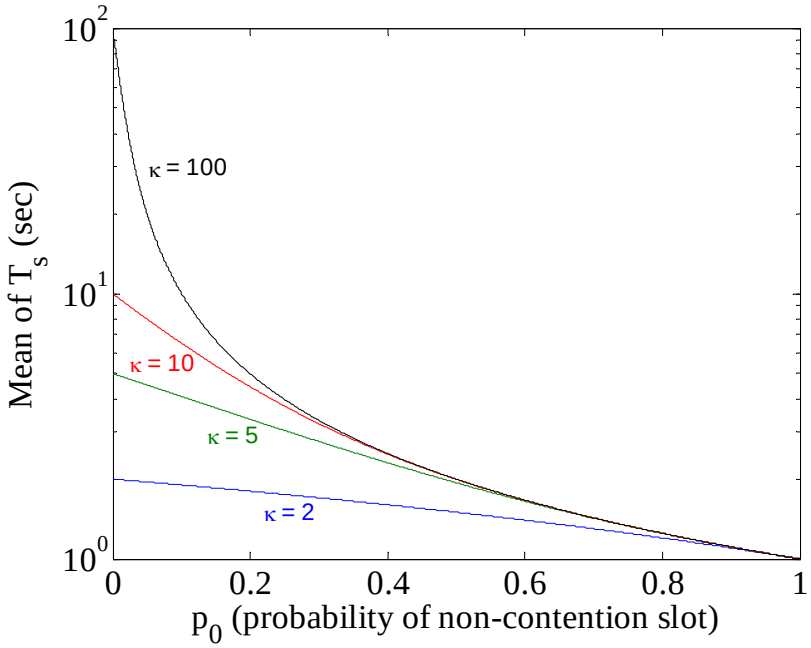


Figure 40. The mean of T_s^{nc} as a function of p_0 for various values of κ with $t_f = 1$ sec.

The second moment of the effective service time is defined as

$$E\left[\left(T_s^{nc}\right)^2\right]=\sum_{i=1}^{\infty}\left(T_{s_i}^{nc}\right)^2\times \Pr[T_s^{nc}=T_{s_i}^{nc}], \tag{106}$$

which, again from (93), is equivalent to

$$E\left[\left(T_s^{nc}\right)^2\right]=\sum_{i=1}^{\kappa-1}\left(it_f\right)^2 p_0\left(1-p_0\right)^{i-1}+\left(\kappa t_f\right)^2\left(1-p_0\right)^{\kappa-1} \tag{107}$$

after the appropriate substitutions. Rearranging, we have

$$E\left[\left(T_s^{nc}\right)^2\right]=\left(t_f\right)^2 p_0\sum_{i=1}^{\kappa-1} i^2\left(1-p_0\right)^{i-1}+\left(\kappa t_f\right)^2\left(1-p_0\right)^{\kappa-1}. \tag{108}$$

To evaluate the summation in the first term, we multiply (99) by c and again take the partial derivative,

$$\frac{\delta}{\delta c}c\left(\sum_{i=0}^{n} ic^{i-1}=\frac{(n+1)c^n(c-1)-\left(c^{n+1}-1\right)}{(c-1)^2}\right). \tag{109}$$

Rearranging,

$$\sum_{i=0}^{n} i\frac{\delta}{\delta c}c^i=\frac{\delta}{\delta c}\left(\frac{(n+1)c^{n+1}(c-1)-c\left(c^{n+1}-1\right)}{(c-1)^2}\right). \tag{110}$$

Differentiating the left side and multiplying out the right side,

$$\sum_{i=0}^{n} i^2c^{i-1}=\frac{\delta}{\delta c}\left(\frac{(n+1)c^{n+2}-(n+1)c^{n+1}-c^{n+2}+c}{(c-1)^2}\right). \tag{111}$$

Combining like terms,

$$\sum_{i=0}^{n} i^2c^{i-1}=\frac{\delta}{\delta c}\left(\frac{nc^{n+2}-(n+1)c^{n+1}+c}{(c-1)^2}\right) \tag{112}$$

and differentiating the right side,

$$\sum_{i=0}^{n} i^2c^{i-1}=\frac{\left((n+2)nc^{n+1}-(n+1)^2c^n+1\right)(c-1)^2-\left(nc^{n+2}-(n+1)c^{n+1}+c\right)2(c-1)}{(c-1)^4}. \tag{113}$$

Reducing,

$$\sum_{i=0}^{n} i^2c^{i-1}=\frac{\left((n+2)nc^{n+1}-(n+1)^2c^n+1\right)(c-1)-2\left(nc^{n+2}-(n+1)c^{n+1}+c\right)}{(c-1)^3} \tag{114}$$

and expanding the individual terms,

$$\sum_{i=0}^{n} i^2 c^{i-1} = \frac{1}{(c-1)^3}\left\{\left(n^2+2n\right)c^{n+2} - \left(n^2+2n+1\right)c^{n+1} + c - \left(n^2+2n\right)c^{n+1} + (n+1)^2 c^n - 1 - 2nc^{n+2} + 2(n+1)c^{n+1} - 2c\right\}. \tag{115}$$

Combining terms,

$$\sum_{i=0}^{n} i^2 c^{i-1} = \frac{1}{(c-1)^3}\left\{n^2 c^{n+2} - \left(2n^2+2n-1\right)c^{n+1} + (n+1)^2 c^n - c - 1\right\}. \tag{116}$$

Again making the substitutions $n = \kappa - 1$ and $c = 1 - p_0$,

$$\sum_{i=0}^{\kappa-1} i^2 (1-p_0)^{i-1} = \frac{1}{(1-p_0-1)^3}\left\{(\kappa-1)^2 (1-p_0)^{\kappa-1+2} - \left(2(\kappa-1)^2 + 2(\kappa-1) - 1\right)(1-p_0)^{\kappa-1+1} + (\kappa-1+1)^2 (1-p_0)^{\kappa-1} - (1-p_0) - 1\right\} \tag{117}$$

and reducing,

$$\sum_{i=0}^{\kappa-1} i^2 (1-p_0)^{i-1} = \frac{1}{(-p_0)^3}\left\{(\kappa-1)^2 (1-p_0)^{\kappa+1} - \left(2(\kappa-1)^2 + 2(\kappa-1) - 1\right)(1-p_0)^{\kappa} + (\kappa)^2 (1-p_0)^{\kappa-1} - (1-p_0) - 1\right\}. \tag{118}$$

Combining terms and moving the negative into the bracketed term,

$$\sum_{i=0}^{\kappa-1} i^2 (1-p_0)^{i-1} = \frac{1}{(p_0)^3}\left\{1 - (1-p_0)[(\kappa-1)^2 (1-p_0)^{\kappa} - \left(2\kappa^2 - 2\kappa - 1\right)(1-p_0)^{\kappa-1} + \kappa^2 (1-p_0)^{\kappa-2} - 1]\right\}. \tag{119}$$

Substituting (119) into (108),

$$E\left[\left(T_s^{nc}\right)^2\right] = \left(t_f\right)^2 p_0 \frac{1}{(p_0)^3}\left\{1 - (1-p_0)[(\kappa-1)^2 (1-p_0)^{\kappa} - \left(2\kappa^2 - 2\kappa - 1\right)(1-p_0)^{\kappa-1} + \kappa^2 (1-p_0)^{\kappa-2} - 1]\} + \left(\kappa t_f\right)^2 (1-p_0)^{\kappa-1}\right\} \tag{120}$$

and pulling out the $\left(\frac{t_f}{p_0}\right)^2$ factor,

$$E\left[\left(T_s^{nc}\right)^2\right] = \left(\frac{t_f}{p_0}\right)^2 \left\{1 - (1-p_0)[(\kappa-1)^2 (1-p_0)^{\kappa} - \left(2\kappa^2 - 2\kappa - 1\right)(1-p_0)^{\kappa-1} + \kappa^2 (1-p_0)^{\kappa-2} - 1] + \kappa^2 (p_0)^2 (1-p_0)^{\kappa-1}\right\}. \tag{121}$$

Combining terms, we finally arrive at the second moment of T_s^{nc} as

$$E\left[\left(T_s^{nc}\right)^2\right]=\left(\frac{t_f}{p_0}\right)^2\left\{1-(1-p_0)[(\kappa-1)^2(1-p_0)^{\kappa}-(2\kappa^2-2\kappa-1)(1-p_0)^{\kappa-1} +\kappa^2\left(1-(p_0)^2\right)(1-p_0)^{\kappa-2}-1]\right\}. \tag{122}$$

The variance of the effective service time is defined as the square of the mean subtracted from the second moment or

$$VAR\left[\left(T_s^{nc}\right)\right]=E\left[\left(T_s^{nc}\right)^2\right]-\left(\overline{T}_s^{nc}\right)^2. \tag{123}$$

Substituting (105) and (122) into (123), we have

$$VAR\left[\left(T_s^{nc}\right)\right]=\left(\frac{t_f}{p_0}\right)^2\left\{1-(1-p_0)[(\kappa-1)^2(1-p_0)^{\kappa}-(2\kappa^2-2\kappa-1)(1-p_0)^{\kappa-1} +\kappa^2\left(1-(p_0)^2\right)(1-p_0)^{\kappa-2}-1]\right\} -\left(\frac{t_f}{p_0}\right)^2\left(1-(1-p_0)^{\kappa}\right)^2. \tag{124}$$

Factoring out the $\left(\frac{t_f}{p_0}\right)^2$ term and expanding,

$$VAR\left[\left(T_s^{nc}\right)\right]=\left(\frac{t_f}{p_0}\right)^2\left\{1-(1-p_0)[(\kappa-1)^2(1-p_0)^{\kappa}-(2\kappa^2-2\kappa-1)(1-p_0)^{\kappa-1} +\kappa^2\left(1-(p_0)^2\right)(1-p_0)^{\kappa-2}-1] -\left(1-2(1-p_0)^{\kappa}+(1-p_0)^{2\kappa}\right)\right\}. \tag{125}$$

Combining terms, we finally arrive at

$$VAR\left[\left(T_s^{nc}\right)\right]=\left(\frac{t_f}{p_0}\right)^2(1-p_0)\left\{(1-p_0)^{2\kappa}+\left((\kappa-1)^2-1\right)(1-p_0)^{\kappa} -(2\kappa^2-2\kappa-1)(1-p_0)^{\kappa-1} +\kappa^2\left(1-(p_0)^2\right)(1-p_0)^{\kappa-2}-1\right\}. \tag{126}$$

Checking the limiting cases of $p_0=1$ (all non-contention slots) and $p_0=0$ (all contention slots), we find that the variance of T_s^{nc} is zero for both cases. This indicates that they are, as expected, deterministic. In the $p_0=0$ case, this result is arrived at

through two applications of L'Hôpital's Rule. A plot of the variance of T_s^{nc} as a function of p_0 for various values of κ is provided in Figure 41.

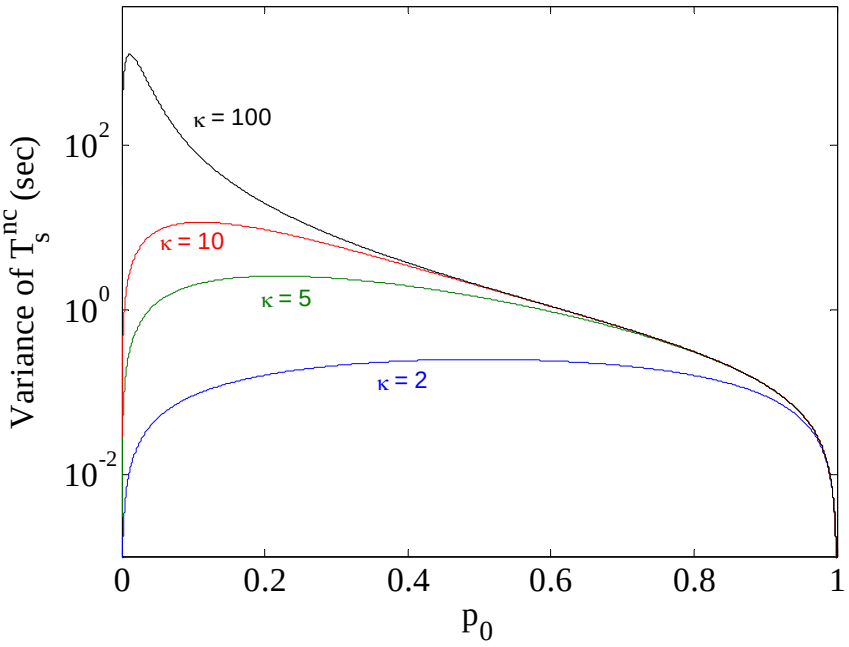


Figure 41. Variance of T_s^{nc} as a function of p_0 for various values of κ with $t_f = 1$ sec.

We can now make use of the well-known mean waiting time result for an M/G/1 queue [41],

$$\bar{T}_w = \frac{\rho \bar{T}_s}{2(1-\rho)}\left(1+\left(\frac{VAR\left[(T_s)\right]}{(\bar{T}_s)^2}\right)\right) \tag{127}$$

where $\rho = \lambda \bar{T}_s < 1$. The latter term in (127), the variance over the mean squared for the service time, is often referred to as the square of the coefficient of variation and, in our case, from (105) and (126), is

$$\frac{VAR\left[\left(T_s^{nc}\right)\right]}{\left(\overline{T}_s^{nc}\right)^2}=\left(\frac{1-p_0}{\left(1-\left(1-p_0\right)^{\kappa}\right)^2}\right)\left\{\left(1-p_0\right)^{2\kappa}+\left(\left(\kappa-1\right)^2-1\right)\left(1-p_0\right)^{\kappa}\right.$$
$$-\left(2\kappa^2-2\kappa-1\right)\left(1-p_0\right)^{\kappa-1} \quad . \qquad (128)$$
$$\left.+\kappa^2\left(1-\left(p_0\right)^2\right)\left(1-p_0\right)^{\kappa-2}-1\right\}.$$

A value of zero for the coefficient of variation indicates that the service time is deterministic while a value of one indicates that it is exponential [41]. Since the variance is zero and the mean of the square is non-zero in the limiting cases of $p_0=1$ and $p_0=0$, the coefficient of variation in (128) is zero in both cases, as expected. A plot of the coefficient of variation of T_s^{nc} as a function of p_0 for various values of κ is provided in Figure 42. Interestingly, the coefficient of variation is less than zero and approaches one as p_0 becomes small and κ becomes large. Thus, the distribution of the service time approaches exponential as the probability of a non-contention slot decreases and ratio of T_{data} for the non-contention slot to that for a contention slot increases.

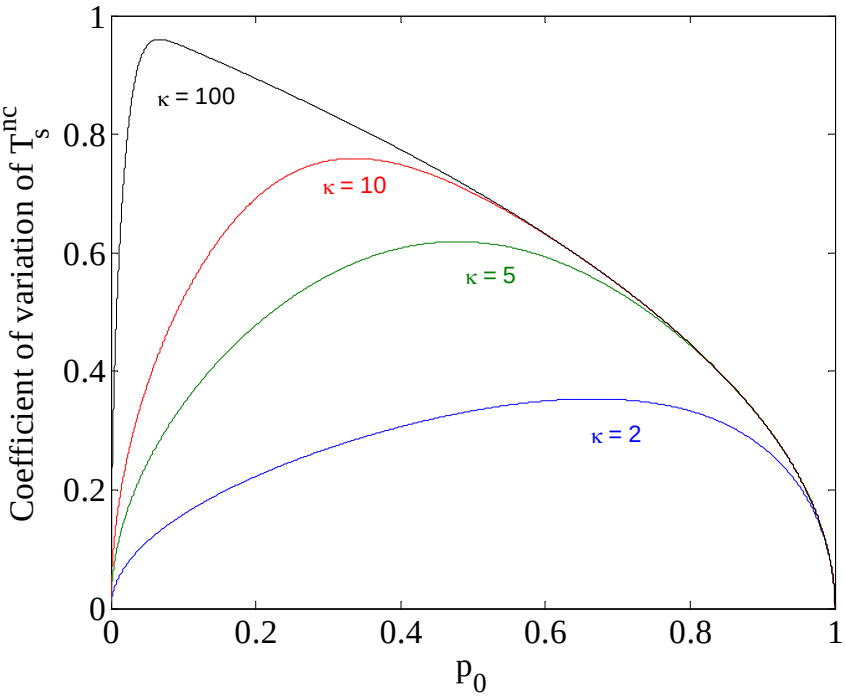


Figure 42. Coefficient of variation of T_s^{nc} as a function of p_0 for various values of κ.

Finally, from (104), we can substitute

$$\rho_{nc} = \lambda_{nc}\bar{T}_s^{nc} = \lambda_{nc}\left(\frac{1-(1-p_0)^{\kappa}}{p_0}\right) < 1 \tag{129}$$

into (127) to arrive at

$$\bar{T}_w^{nc} = \frac{\lambda_{nc}\left(\frac{1-(1-p_0)^{\kappa}}{p_0}\right)^2}{2\left(1-\lambda_{nc}\left(\frac{1-(1-p_0)^{\kappa}}{p_0}\right)\right)}\left(1+\left(\frac{VAR\left[(T_s)\right]}{(\bar{T}_s)^2}\right)\right). \tag{130}$$

Again, looking at the limiting cases, we find that

$$\begin{aligned} \lim_{p_0\to 1}\bar{T}_w^{nc} &= \frac{\lambda_{nc}(t_f)^2}{2(1-\lambda_{nc}t_f)} \quad \text{and} \\ \lim_{p_0\to 0}\bar{T}_w^{nc} &= \frac{\lambda_{nc}(\kappa t_f)^2}{2(1-\lambda_{nc}\kappa t_f)}, \end{aligned} \tag{131}$$

which agree with the deterministic case solved by Lam in [26]. A plot of $\bar{T}_w^{nc}$ as a function of p_0 for various values of κ is provided in Figure 43, while a plot of $\bar{T}_w^{nc}$ as a function of ρ_{nc} for various values of p_0 is provided in Figure 44.

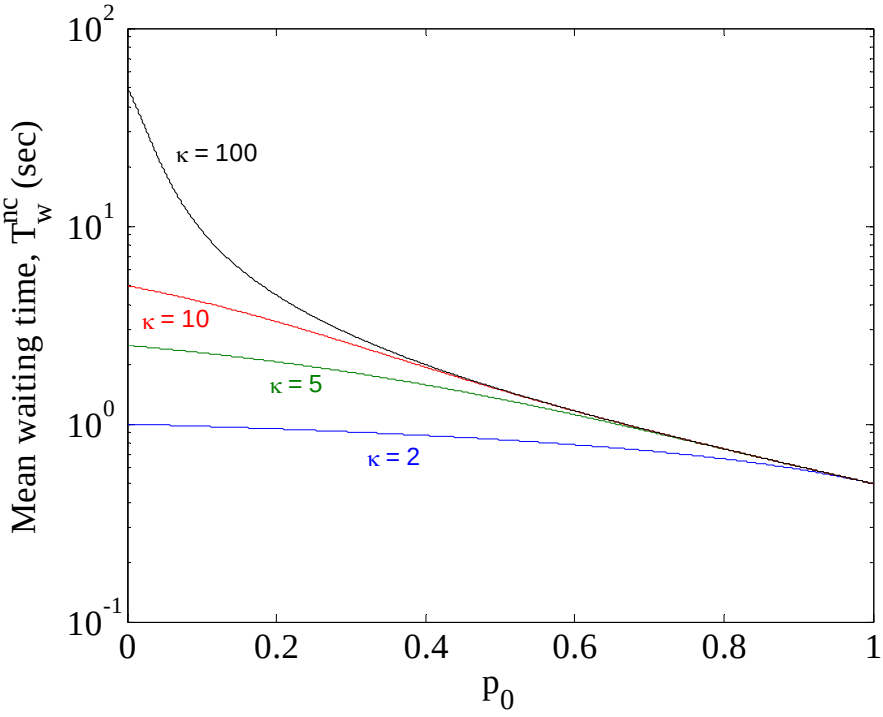


Figure 43. $\bar{T}_w^{nc}$ as a function of p_0 for various values of κ with $t_f = 1$ sec.

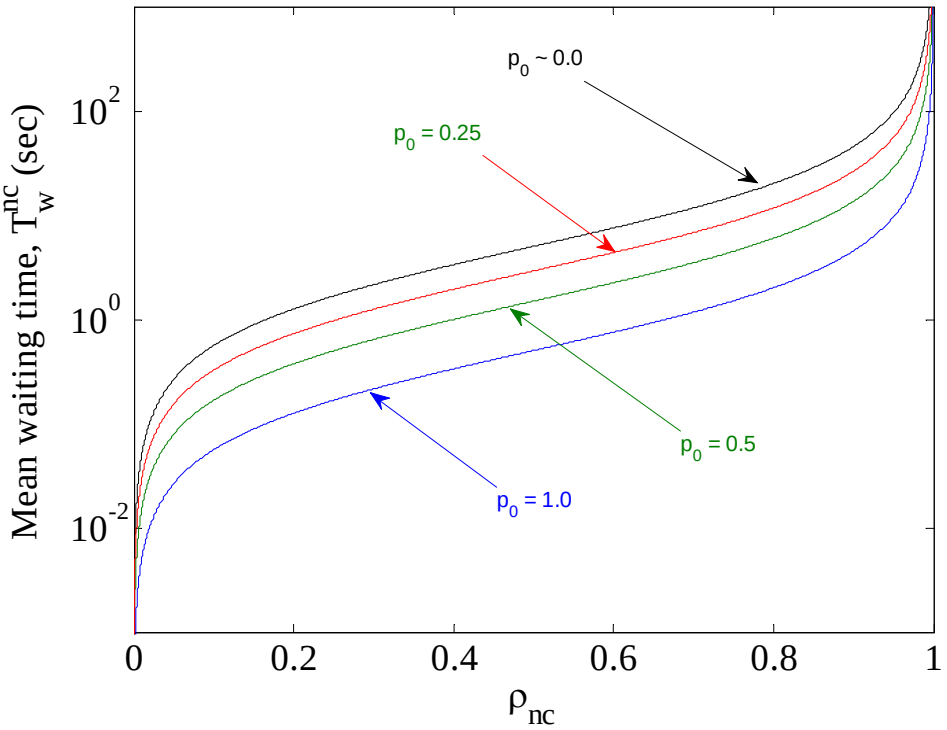


Figure 44. $\bar{T}_w^{nc}$ as a function of ρ_{nc} for various values of p_0 with $t_f = 1$ sec.

Turning our attention to $\overline{T}_{xmt}^{nc}$, this is simply the mean time required to transmit a packet once its waiting time is complete. We begin by calculating $\overline{T}_{xmt}^{nc}$ for a packet that is transmitted in a single non-contention slot, denoted $T_{xmt_1}^{nc}$. Again assuming that exactly one non-contention packet is transmitted during a non-contention slot, the transmission time for a packet transmitted in a non-contention slot is, from (71),

$$T_{xmt_1}^{nc} = t_{IFS} + t_{nc} \tag{132}$$

where t_{IFS} is included to account for the delay between the slot boundary and the actual start of the transmission. For a packet that is transmitted in a contention slot, the actual transmission will span across multiple frames, as discussed above. Examining the case where a packet is transmitted in a contention slot followed by a non-contention slot, denoted $T_{xmt_2}^{nc}$,we have, again from (71),

$$T_{xmt_2}^{nc} = t_f + \left(t_{IFS} + t_{nc}\right) - \left(t_{nc} + t_{IFS} - t_b - kt_{ms}\right) \tag{133}$$

where, as in (132), the first term t_f accounts for the single frame time to get to the second (non-contention) slot, the second term $\left(t_{IFS} + t_{nc}\right)$ accounts for the packet transmission in this final non-contention slot, and the third term $\left(t_{nc} + t_{IFS} - t_b - kt_{ms}\right)$ reduces the transmission time required in this final non-contention slot by the amount of the packet that was transmitted in the prior contention slot. Extending this argument to the general case in which we have $\alpha - 1$ consecutive contention slots followed by a non-contention slot, where $\alpha \le \kappa$ in (87),

$$T_{xmt_\alpha}^{nc} = (\alpha - 1)t_f + \left(t_{IFS} + t_{nc}\right) - (\alpha - 1)\left(t_{nc} + t_{IFS} - t_b - kt_{ms}\right). \tag{134}$$

Following the development of (88), the probability $T_{xmt_\alpha}^{nc}$ is a function of both α and p_0 and is given by

$$\Pr\left[T_{xmt}^{nc} = T_{xmt_\alpha}^{nc}\right] = p_0\left(1 - p_0\right)^{\alpha - 1} \qquad \text{for } 1 \le \alpha \le \kappa. \tag{135}$$

Here, as opposed to (88), we have included κ in the range of α because we must explicitly account for the case of $\kappa - 1$ consecutive contention slots followed by a non-contention slot as well as the case of κ consecutive contention slots. For this latter case, we have

$$T_{xmt_{\kappa c}}^{nc} = (\kappa - 1)t_f + (t_b + kt_{ms}) + \left[t_{nc} - (\kappa - 1)t_{nc2}\right] \tag{136}$$

where the second term $(t_b + kt_{ms})$ now accounts for the fact that the final slot is a contention slot and we have defined $t_{nc2} = t_{nc} + t_{IFS} - t_b - kt_{ms}$. The probability of this case is

$$\Pr\left[T_{xmt}^{nc} = T_{xmt_{\kappa c}}^{nc}\right] = (1 - p_0)^{\kappa}. \tag{137}$$

We can calculate the mean transmission time $\bar{T}_{xmt}^{nc}$ by combining (134) through (137), which, after a little algebraic manipulation, is

$$\begin{aligned} \bar{T}_{xmt}^{nc} = &\sum_{i=1}^{\kappa}\left[(i-1)(t_f - t_{nc2}) + t_{IFS} + t_{nc}\right]p_0(1-p_0)^{i-1} \\ &+\left[(\kappa-1)(t_f - t_{nc2}) + t_b + kt_{ms} + t_{nc}\right](1-p_0)^{\kappa}. \end{aligned} \tag{138}$$

Rearranging terms,

$$\begin{aligned} \bar{T}_{xmt}^{nc} = &\sum_{i=1}^{\kappa}\left[i(t_f - t_{nc2}) - (t_f - (t_{IFS} + t_{nc} + t_{nc2}))\right]p_0(1-p_0)^{i-1} \\ &+\left[(\kappa-1)(t_f - t_{nc2}) + t_b + kt_{ms} + t_{nc}\right](1-p_0)^{\kappa} \end{aligned} \tag{139}$$

and distributing,

$$\begin{aligned} \bar{T}_{xmt}^{nc} = &p_0\sum_{i=1}^{\kappa}\left[i(t_f - t_{nc2})(1-p_0)^{i-1} - (t_f - (t_{IFS} + t_{nc} + t_{nc2}))(1-p_0)^{i-1}\right] \\ &+\left[(\kappa-1)(t_f - t_{nc2}) + t_b + kt_{ms} + t_{nc}\right](1-p_0)^{\kappa}. \end{aligned} \tag{140}$$

Pulling the appropriate terms out of the summation and distributing it across,

$$\begin{aligned} \bar{T}_{xmt}^{nc} = &(t_f - t_{nc2})\,p_0\sum_{i=1}^{\kappa} i(1-p_0)^{i-1} \\ &-(t_f - (t_{IFS} + t_{nc} + t_{nc2}))\,p_0\sum_{i=1}^{\kappa}(1-p_0)^{i-1} \\ &+\left[(\kappa-1)(t_f - t_{nc2}) + t_b + kt_{ms} + t_{nc}\right](1-p_0)^{\kappa}. \end{aligned} \tag{141}$$

Adjusting the indices on the summations,

$$\begin{aligned}\bar{T}_{xmt}^{nc} &= \left(t_f - t_{nc2}\right) p_0 \sum_{i=0}^{\kappa} i\left(1-p_0\right)^{i-1} \\ &\quad -\left(t_f - \left(t_{IFS} + t_{nc} + t_{nc2}\right)\right) p_0 \sum_{j=0}^{\kappa-1} \left(1-p_0\right)^{j} \\ &\quad +\left[\left(\kappa-1\right)\left(t_f - t_{nc2}\right) + t_b + kt_{ms} + t_{nc}\right]\left(1-p_0\right)^{\kappa}\end{aligned} \tag{142}$$

and using (91), (116) and the appropriate substitutions, we have

$$\begin{aligned}\bar{T}_{xmt}^{nc} &= \left(t_f - t_{nc2}\right) p_0 \frac{\left(\kappa+1\right)\left(1-p_0\right)^{\kappa}\left(\left(1-p_0\right)-1\right) - \left(\left(1-p_0\right)^{\kappa+1}-1\right)}{\left(\left(1-p_0\right)-1\right)^2} \\ &\quad -\left(t_f - \left(t_{IFS} + t_{nc} + t_{nc2}\right)\right) p_0 \frac{\left(1-p_0\right)^{\kappa-1+1}-1}{1-p_0-1} \\ &\quad +\left[\left(\kappa-1\right)\left(t_f - t_{nc2}\right) + t_b + kt_{ms} + t_{nc}\right]\left(1-p_0\right)^{\kappa}.\end{aligned} \tag{143}$$

Simplifying and rearranging terms,

$$\begin{aligned}\bar{T}_{xmt}^{nc} &= \left(t_f - t_{nc2}\right) \frac{\left(\kappa+1\right)\left(1-p_0\right)^{\kappa+1} - \left(\kappa+1\right)\left(1-p_0\right)^{\kappa} - \left(1-p_0\right)^{\kappa+1} + 1}{p_0} \\ &\quad -\left(t_f - \left(t_{IFS} + t_{nc} + t_{nc2}\right)\right) p_0 \frac{\left(1-p_0\right)^{\kappa}-1}{\left(-p_0\right)} \\ &\quad +\left[\left(\kappa-1\right)\left(t_f - t_{nc2}\right) + t_b + kt_{ms} + t_{nc}\right]\left(1-p_0\right)^{\kappa}\end{aligned} \tag{144}$$

and further algebraic manipulation leads to

$$\begin{aligned}\bar{T}_{xmt}^{nc} &= \left[t_f - t_{nc2}\right]\left(\frac{\kappa\left(1-p_0\right)^{\kappa+1} - \left(\kappa+1\right)\left(1-p_0\right)^{\kappa} + 1}{p_0}\right) \\ &\quad -\left[t_f - \left(t_{IFS} + t_{nc} + t_{nc2}\right)\right]\left(1-\left(1-p_0\right)^{\kappa}\right) \\ &\quad +\left[\left(\kappa-1\right)\left(t_f - t_{nc2}\right) + t_b + kt_{ms} + t_{nc}\right]\left(1-p_0\right)^{\kappa}.\end{aligned} \tag{145}$$

Expanding,

$$\begin{aligned}\bar{T}_{xmt}^{nc} &= \left[t_f - t_{nc2}\right]\left(\frac{\kappa\left(1-p_0\right)^{\kappa+1} - \left(\kappa+1\right)\left(1-p_0\right)^{\kappa} + 1}{p_0}\right) \\ &\quad +\left[\kappa\left(t_f - t_{nc2}\right) - t_f + t_{nc2} + t_b + kt_{ms} + t_{nc} + t_f - t_{IFS} - t_{nc} - t_{nc2}\right]\left(1-p_0\right)^{\kappa} \\ &\quad -t_f + t_{IFS} + t_{nc} + t_{nc2}\end{aligned} \tag{146}$$

and, finally, simplifying to arrive at

$$\bar{T}_{xmt}^{nc} = \left[t_f - t_{nc2}\right]\left(\frac{\kappa(1-p_0)^{\kappa+1} - (\kappa+1)(1-p_0)^{\kappa} + 1}{p_0}\right) + \left[\kappa(t_f - t_{nc2}) - t_{IFS} + t_b + kt_{ms}\right](1-p_0)^{\kappa} - t_f + t_{IFS} + t_{nc} + t_{nc2}. \quad (147)$$

Checking the limiting cases of $p_0 = 1$ (all non-contention slots) and $p_0 = 0$ (all contention slots), we obtain the expected results from (132) and (136) of

$$\lim_{p_0 \to 1} \bar{T}_{xmt}^{nc} = t_{IFS} + t_{nc} \quad \text{and}$$
$$\lim_{p_0 \to 0} \bar{T}_{xmt}^{nc} = (\kappa - 1)t_f + t_b + \kappa t_{ms} + \left[t_{nc} - (\kappa - 1)t_{nc2}\right]. \quad (148)$$

The latter result is arrived at through a single application of L'Hôpital's Rule. A plot of the mean transmission time as a function of p_0 for various values of κ is provided in Figure 45.

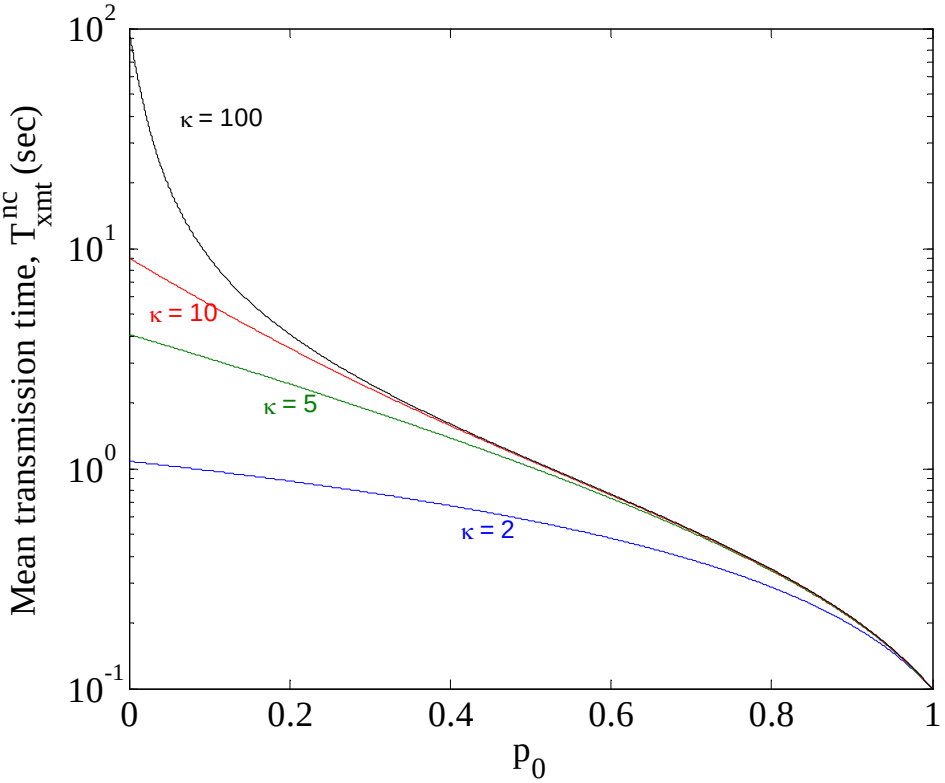


Figure 45. The mean transmission time as a function of p_0 for various values of κ with $t_f = 1$ sec.

We can finally arrive at an expression for the total mean packet delay for the contention mode by substituting (85), (130), and (147) into (84),

$$\begin{aligned}\bar{D}^{nc} = \frac{t_s}{2} + \frac{\lambda_{nc}\left(\frac{1-(1-p_0)^{\kappa}}{p_0}\right)^2}{2\left(1-\lambda_{nc}\left(\frac{1-(1-p_0)^{\kappa}}{p_0}\right)\right)}\left(1+\left(\frac{VAR\left[(T_s)\right]}{(\bar{T}_s)^2}\right)\right) \\ +\left[t_f - t_{nc2}\right]\left(\frac{\kappa(1-p_0)^{\kappa+1}-(\kappa+1)(1-p_0)^{\kappa}+1}{p_0}\right) \\ +\left[\kappa(t_f - t_{nc2}) - t_{IFS} + t_b + kt_{ms}\right](1-p_0)^{\kappa} \\ -t_f + t_{IFS} + t_{nc} + t_{nc2} + t_{prop}\end{aligned} \tag{149}$$

where

$$\begin{aligned}\frac{VAR\left[\left(T_s^{nc}\right)\right]}{\left(\bar{T}_s^{nc}\right)^2} = \left(\frac{1-p_0}{\left(1-(1-p_0)^{\kappa}\right)^2}\right)\Big\{(1-p_0)^{2\kappa} + \left((\kappa-1)^2-1\right)(1-p_0)^{\kappa} \\ -\left(2\kappa^2-2\kappa-1\right)(1-p_0)^{\kappa-1} \\ +\kappa^2\left(1-(p_0)^2\right)(1-p_0)^{\kappa-2}-1\Big\}.\end{aligned} \tag{150}$$

A plot of the mean total packet delay as a function of p_0 for various values of κ is provided in Figure 46, while it is plotted as a function of ρ_{nc} for various values of p_0 in Figure 47. Finally, it is plotted in Figure 48 as a function of the aggregate non-contention load, Λ_{nc}, for various values of p_0.

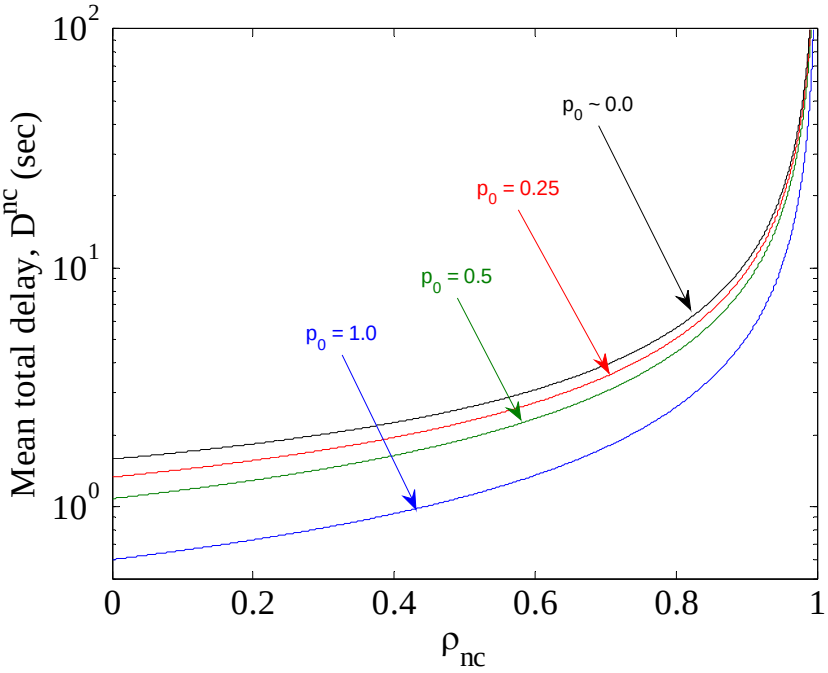


Figure 46. Non-contention mode mean total packet delay as a function of ρ_{nc} for various values of p_0 with $t_f = 1$ sec.

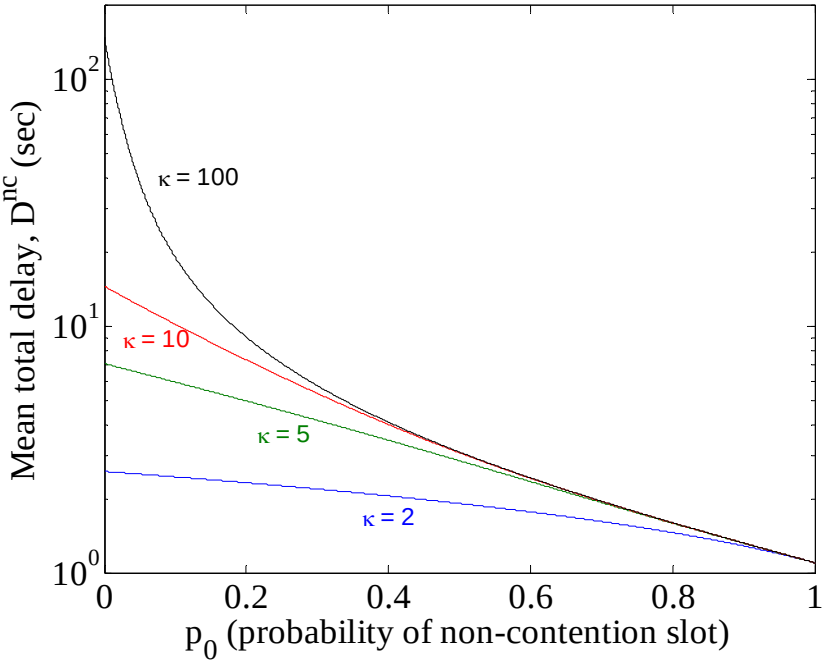


Figure 47. Non-contention mode mean total packet delay as a function of p_0 for various values of κ with $t_f = 1$ sec.

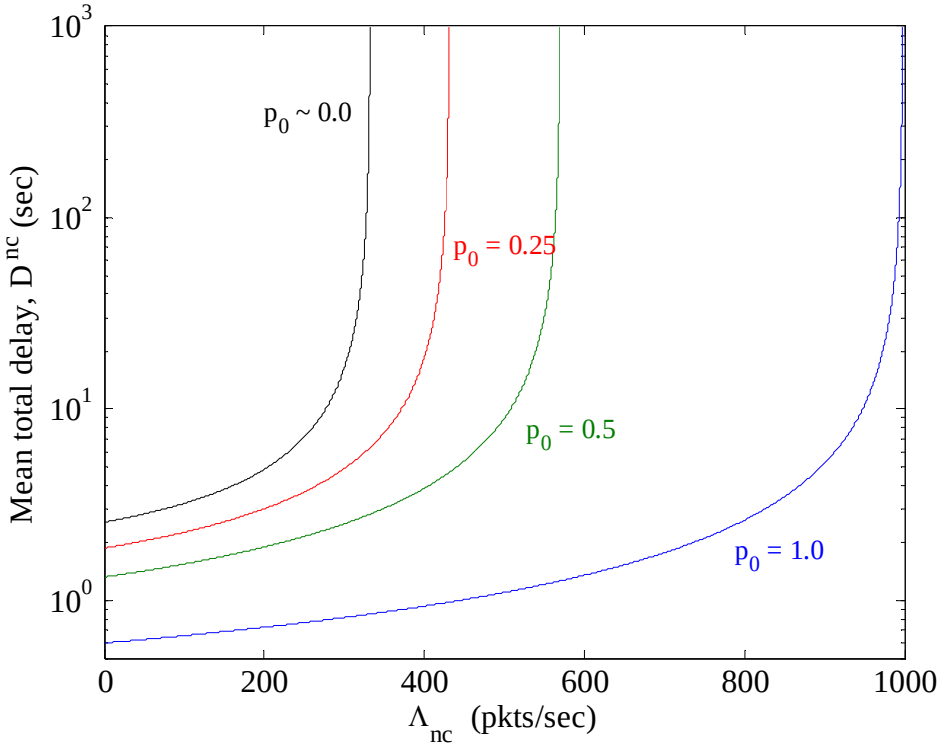


Figure 48. Non-contention mode mean total packet delay as a function of Λ_{nc} (aggregate packet arrival rate) for various values of p_0 with $t_f = 1$ sec.

3. Slotted ALOHA Model with Periodic Server Vacations

We begin the analysis of the contention mode by developing a model for a slotted ALOHA system with periodic server vacations. By this, we mean that the service will be governed by a fixed cycle comprised of alternating active and inactive periods. During the inactive period, the server will shut down and not be available to serve the queued packets. We also make the assumption that once a server has entered an active period, all subsequent packet arrivals will be deferred to the next active period (i.e., a packet must arrive prior to an active period to be eligible for service in that period). We define K as the number of slots in an active period. Following the work of [153]and [154], we make use of the model in Figure 49. Here, a node attempts transmission in a given slot with probability *p* and, if the transmission is unsuccessful, the packet is requeued.

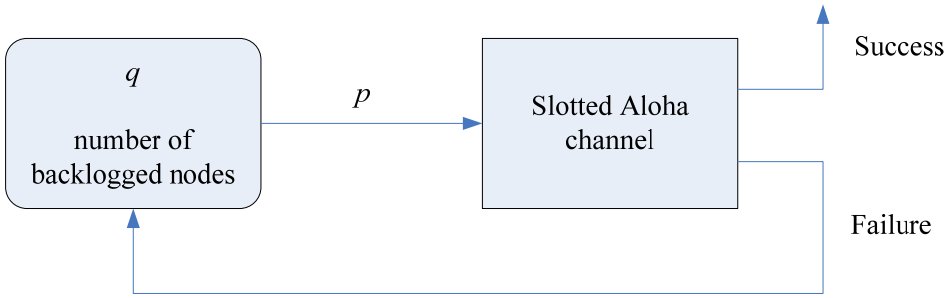


Figure 49. Model of a slotted ALOHA channel with q backlogged nodes (After [153]).

To develop the associated discrete Markov chain, we define a state by the number of nodes with a packet queued for transmission and derive the associated state transition probability matrix, P. We define $p_{i,j}$ as the probability that the system will transition from state i to state j in a given slot. We begin by recognizing that the probability of a transition from i to j where $j > i$ is zero during the active period because we have assumed that all additional arrivals are deferred to the next active period. Furthermore, the probability of a transition where $j < i-1$ is also zero because there can only be at most one successful transmission per slot. The case of $j = i+1$ represents a single successful transmission. This will occur when any one of the i nodes with a packet queued attempts to transmit and all of the other nodes do not. Since a node will attempt a transmission with probability p, this is simply

$$p_{i,i-1} = \binom{i}{1} p(1-p)^{i-1} = ip(1-p)^{i-1}. \tag{151}$$

This leaves the probability that a node will remain in the current state ($j = i$), which is

$$p_{i,i} = 1 - p_{i,i-1} = 1 - ip(1-p)^{i-1}. \tag{152}$$

Combining these, we have

$$p_{i,j} = \begin{cases} 0 & j < i-1 \\ ip(1-p)^{i-1} & j = i-1 \\ 1-ip(1-p)^{i-1} & j = i \\ 0 & j > i \end{cases}. \tag{153}$$

Defining $\sigma = p(1-p)^{i-1}$, the corresponding $(M+1)\times(M+1)$ probability transition matrix is

$$\mathrm{P} = \begin{bmatrix} 1-\sigma & 0 & 0 \\ \sigma & \ddots & 0 \\ 0 & \sigma & 1-\sigma \end{bmatrix} \tag{154}$$

where M is the number of nodes with a packet queued for transmission at the start of the active period.

The state probability vector $\underline{p}(k)$ is defined as

$$\underline{p}(k) = \begin{bmatrix} p_0(k) \\ p_1(k) \\ \vdots \\ p_M(k) \end{bmatrix}^T \tag{155}$$

where $p_i(k)$ is the probability of state i at the end of slot k and X^T is the transpose of X. Thus, for the resulting Markov chain in Figure 50,

$$\begin{aligned} \underline{p}(1) &= \underline{p}(0)\,\mathrm{P}, \\ \underline{p}(2) &= \underline{p}(1)\,\mathrm{P} = \left(\underline{p}(0)\,\mathrm{P}\right)\mathrm{P} = \underline{p}(0)\,\mathrm{P}^2, \end{aligned} \tag{156}$$

and

$$\underline{p}(k) = \underline{p}(0)\,\mathrm{P}^k = \underline{p}(0)\begin{bmatrix} 1-\sigma & 0 & 0 \\ \sigma & \ddots & 0 \\ 0 & \sigma & 1-\sigma \end{bmatrix}^k \tag{157}$$

where $\underline{p}(0)$ is the initial state probability vector at the start of the active period. Note that the mean number of nodes with packets queued in a given slot k, denoted $Q(k)$, is simply the mean of the appropriate state probability vector as in

$$Q(k) = \underline{p}(k)\begin{bmatrix} 0 \\ 1 \\ \vdots \\ M \end{bmatrix} = \underline{p}(0)\begin{bmatrix} 1-\sigma & 0 & 0 \\ \sigma & \ddots & 0 \\ 0 & \sigma & 1-\sigma \end{bmatrix}^k \begin{bmatrix} 0 \\ 1 \\ \vdots \\ M \end{bmatrix}. \tag{158}$$

For an active period of K slots, the state probability vector at the end of the active period is given by

$$\underline{p}(K) = \underline{p}(0)\,\mathrm{P}^K = \underline{p}(0)\begin{bmatrix} 1-\sigma & 0 & 0 \\ \sigma & \ddots & 0 \\ 0 & \sigma & 1-\sigma \end{bmatrix}^K . \tag{159}$$

and the mean number of nodes with packets queued at the end of an active period is $Q(K)$.

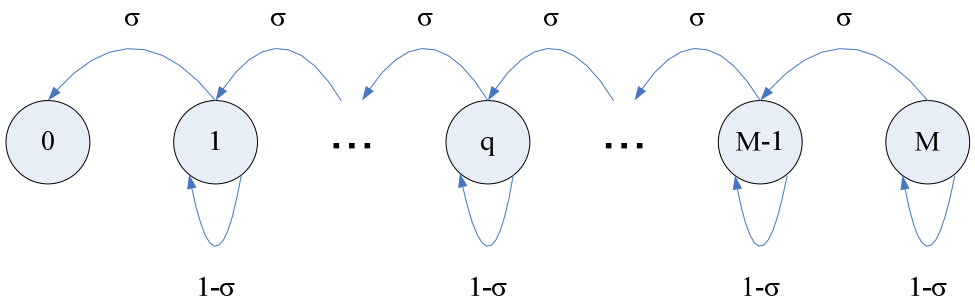


Figure 50. Markov chain for Slotted ALOHA with server vacations.

For a cycle time (defined as one active period followed by one inactive period) of duration T_{cycle} and an arrival rate of λ, the initial mean number of nodes with packets queued at the start of the next active period, denoted $Q'(0)$, is

$$Q'(0) = Q(K) + \lambda T_{cycle} . \tag{160}$$

We then define steady state as the condition where the state probability vector in the next active period is equivalent to the state probability vector in the current active period (i.e., $\underline{p}'(k) = \underline{p}(k)$), which implies that

$$Q'(0) = Q(0),\ \ Q'(K) = Q(K) \tag{161}$$

and, from (160),

$$Q(0) - Q(K) = \lambda T_{cycle} . \tag{162}$$

We now use the results from this model to derive the throughput and delay for slotted ALOHA with periodic server vacations.

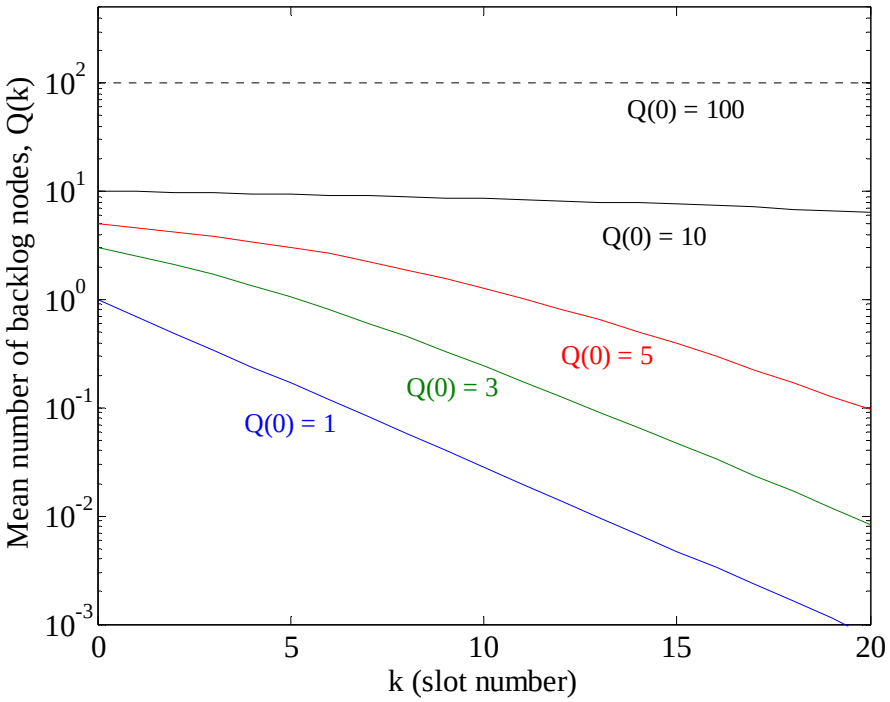


Figure 51. Mean number of backlogged nodes (nodes with a packet queued for transmission) as a function of slot number for various initial state conditions. For this plot, the probability of transmission in a slot, *p*, is 0.3.

4. Throughput for Slotted ALOHA with Periodic Server Vacations

The throughput for slotted ALOHA with periodic server vacations is then simply the difference between the mean number of nodes with packets queued at the beginning of an active period and the mean number at the end of the active period divided by the cycle time, or

$$\text{Throughput} = \frac{Q(0) - Q(K)}{T_{cycle}}. \tag{163}$$

From (162), this implies, as expected, that the throughput at steady state equals the arrival rate. To be consistent with the literature, we can normalize (163) by multiplying it by the packet transmission time T_{xmt}^{c} to arrive at

$$S^{c} = \frac{Q(0) - Q(K)}{T_{cycle}} T_{xmt}^{c}. \tag{164}$$

We plot the throughput as a function of the offered load $Q(0)$ in Figure 52 for various values of κ.

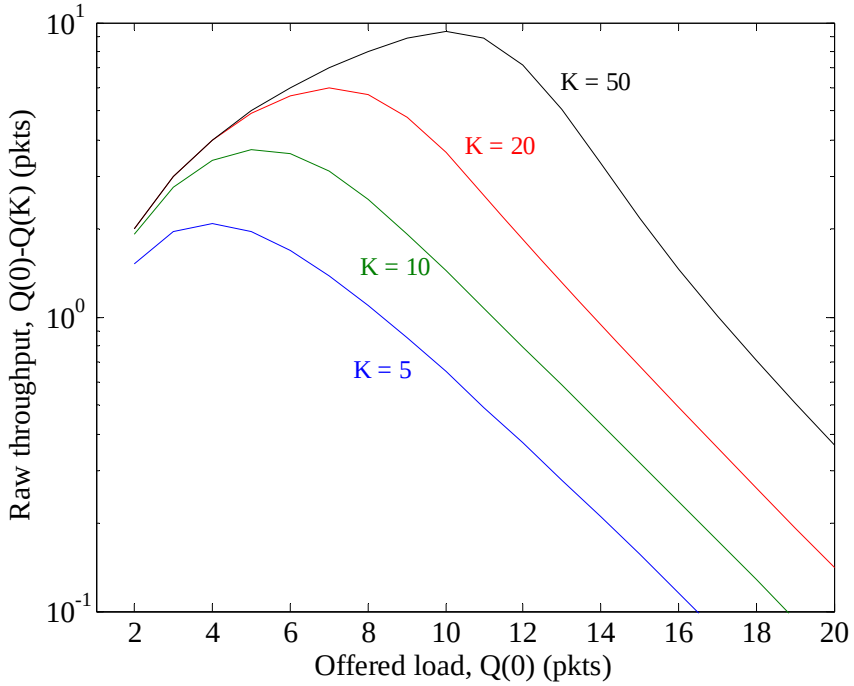


Figure 52. Raw throughput per active period as a function of offered load for various numbers of slots per active period (K). For this plot, the probability of transmission in a slot, p, is 0.3.

5. Delay for Slotted ALOHA with Periodic Server Vacations

As in (84), we can calculate the mean total packet delay for slotted ALOHA with periodic server vacations as the sum of (1) the mean time to synchronize to an active period, (2) the mean waiting time, (3) the mean transmission time and (4) the mean propagation time or

$$\bar{D} = \bar{T}_{sync} + \bar{T}_w + \bar{T}_{xmt} + t_{prop}. \tag{165}$$

The first term can be calculated as in (85) to be

$$\bar{T}_{sync} = \frac{T_{cycle}}{2}. \tag{166}$$

and, because we are assuming that a packet can be transmitted in a single slot, the third term is

$$\bar{T}_{xmt} = \frac{\bar{L}}{R} \tag{167}$$

where $\bar{L}$ is the mean packet length (in bits) and *R* is the channel data rate (bps).

Turning our attention once again to the waiting time, we view the entire system as a single M/G/1 queue and develop the service time distribution using the model developed above. A packet transmitted in the first slot of the active period will experience a service time of T_{xmt} while a packet transmitted in the second slot will wait through the first slot and then transmit resulting in a service time of $T_{slot} + T_{xmt}$. This can be generalized for slot *k* in the active period as

$$T_s(k) = (k-1)T_{slot} + T_{ovhd} + T_{xmt} \tag{168}$$

where we have included the overhead T_{ovhd} in the transmission slot. A packet can also wait across active periods as well. This would occur if the probability of at least one packet queued for transmission at the end of an active period was non-zero. Following the same logic, then, a packet that is transmitted in the m^{th} active period would have to wait an additional $m-1$ cycle times or

$$T_s(m,k) = (m-1)T_{cycle} + (k-1)T_{slot} + T_{ovhd} + T_{xmt}. \tag{169}$$

Thus, the service time is a discrete random variable that can take on the values indicated in (169). To develop the distribution, we must now calculate the probabilities of the discrete values.

The probability that a packet will be successfully transmitted in the first slot of an active period is the probability that one node will transmit and that the remaining $Q(0)-1$ nodes will not. Since a node transmits in a slot with probability *p*, this probability is

$$\Pr[T_s = T_s(0,1)] = p(1-p)^{Q(0)-1}. \tag{170}$$

The probability that a packet will successfully be transmitted in the second slot of an active period is the probability that it wasn't successfully transmitted in the first slot multiplied by the probability that it is transmitted in the second slot and none of the other $Q(1)-1$ nodes transmit or

$$\Pr[T_s = T_s(0,2)] = \left(1 - p\left(1-p\right)^{Q(0)-1}\right) p\left(1-p\right)^{Q(1)-1}. \quad (171)$$

Looking at the next slot, we must include the probability that it was not successfully transmitted in either of the first two slots as in

$$\Pr[T_s = T_s(0,3)] = \left[1 - p\left(1-p\right)^{Q(0)-1} - \left(1 - p\left(1-p\right)^{Q(0)-1}\right) p\left(1-p\right)^{Q(1)-1}\right] p\left(1-p\right)^{Q(2)-1} \quad (172)$$

or,

$$\Pr[T_s = T_s(0,3)] = 1 - (\Pr[T_s = T_s(0,1)] + \Pr[T_s = T_s(0,2)]) p\left(1-p\right)^{Q(3)-1}. \quad (173)$$

Extending this to the general case in the first active period, we have

$$\Pr[T_s = T_s(0,k)] = \left(1 - \sum_{i=1}^{k-1} \Pr[T_s = T_s(0,i)]\right) p\left(1-p\right)^{Q(k-1)-1}. \quad (174)$$

Assuming steady state (i.e. $Q'(k) = Q(k)$), we can also extend this across cycles by adding a second summation as in

$$\Pr[T_s = T_s(m,k)] = \left(1 - \sum_{j=0}^{m-1} \sum_{i=1}^{k-1} \Pr[T_s = T_s(j,i)]\right) p\left(1-p\right)^{Q(k-1)-1}. \quad (175)$$

This is a recursive equation and can be solved numerically to some desired level of accuracy. The distribution of T_s is then defined by (169) and (175) and we can also numerically calculate its mean and variance to some desired degree of accuracy. The probability distribution and cumulative distribution function for the service time of slotted ALOHA with periodic server vacations are plotted in Figures 53 through 56.

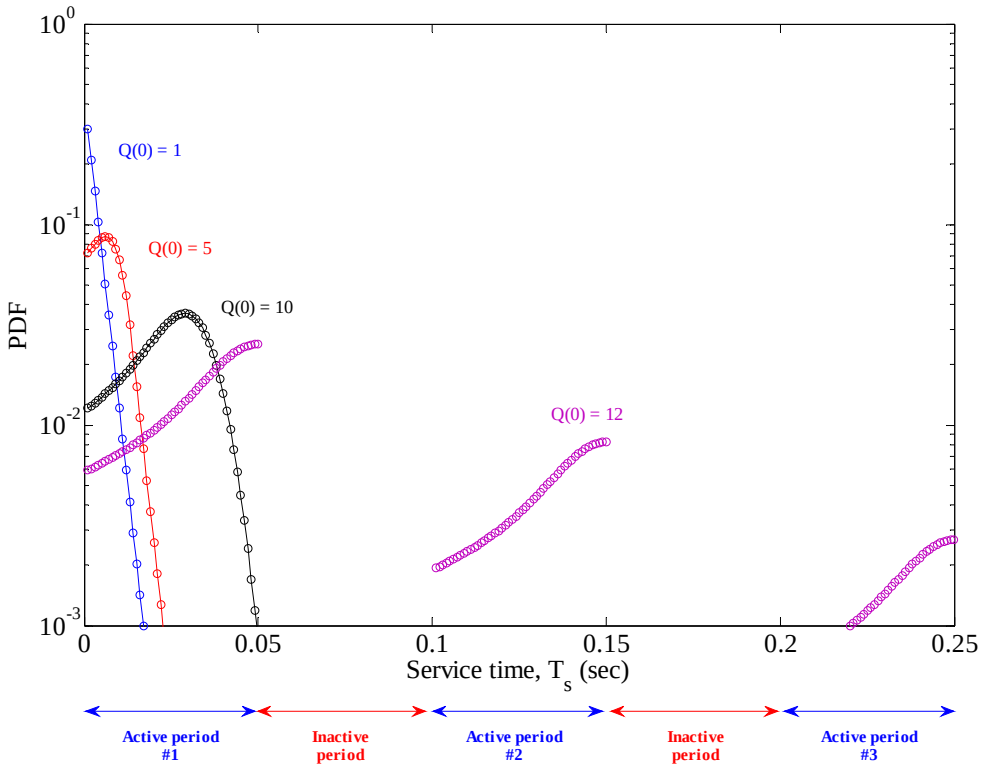


Figure 53. Service time probability distribution in log-linear scale. For this plot, the probability of transmission in a slot, *p*, is 0.3.

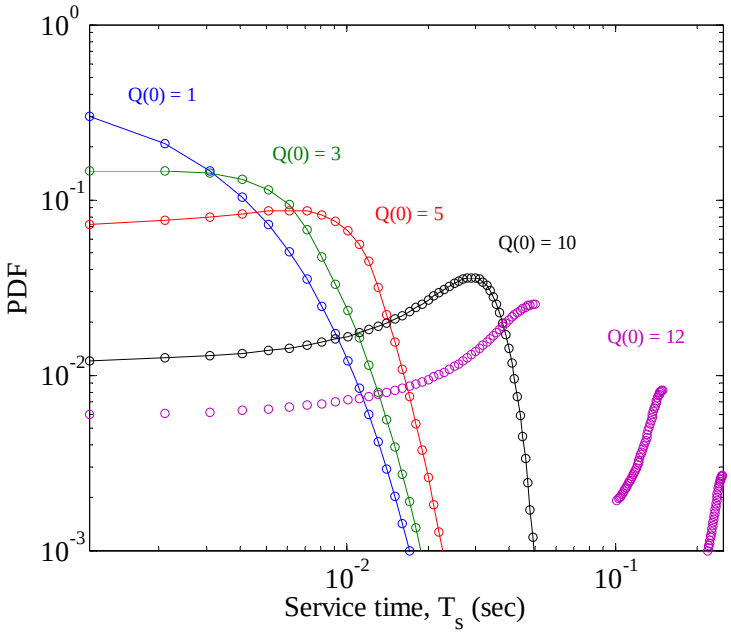


Figure 54. Service time probability distribution in log-log scale. For this plot, the probability of transmission in a slot, *p*, is 0.3.

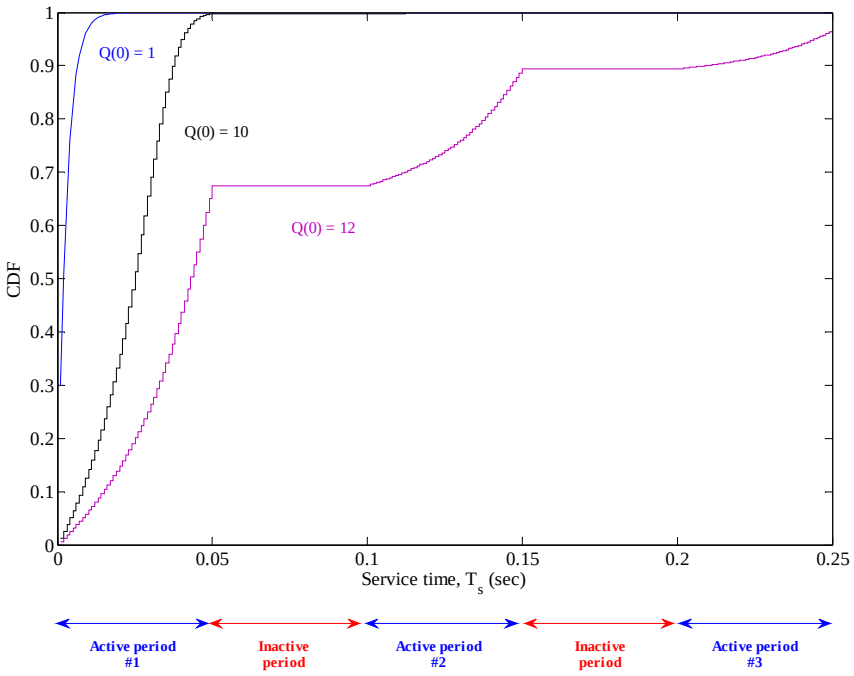


Figure 55. Service time cumulative distribution in linear scale. For this plot, the probability of transmission in a slot, *p*, is 0.3.

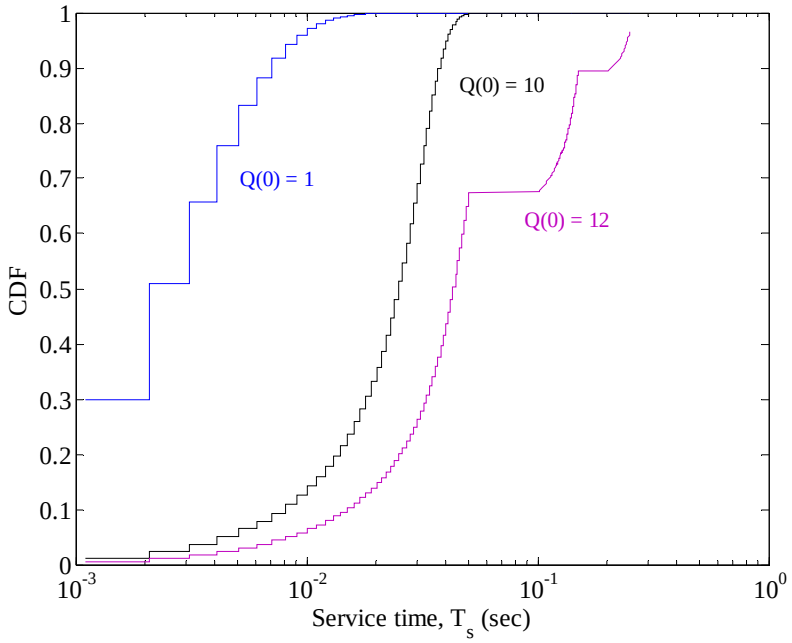


Figure 56. Service time cumulative distribution in linear-log scale. For this plot, the probability of transmission in a slot, *p*, is 0.3.

Given the distribution for the service time, we can now use the M/G/1 waiting time equation of (127) to numerically calculate the waiting for slotted ALOHA with periodic server vacations. Substituting this as well as (166) and (167) into (165), we can then solve for the mean total delay of slotted ALOHA with periodic server vacations. This mean total delay is plotted as a function of packet arrival rate in Figure 57 for various values of *K*.

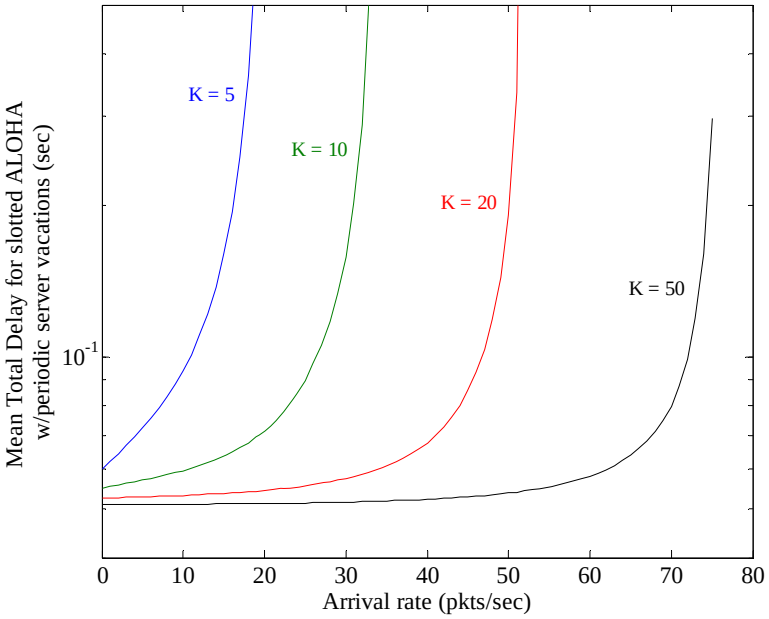


Figure 57. Mean total delay as a function of the packet arrival rate for various numbers of slots in an active period. For this plot, the probability of transmission in a slot, *p*, is 0.3.

6. Contention Throughput and Delay for CWS-MAC

We now use the general results for slotted ALOHA with the periodic server vacations to develop the contention-mode throughput and delay expressions for CWS-MAC. Applying the parameters of CWS-MAC from Figure 27 to the throughput and delay results of the previous two sections, the cycle time is equivalent to the slot time t_s, the duration of the active period is *k* minislots of t_{ms} each, and the overhead within an active period is the beacon time t_b. From (164), the normalized mean throughput for the contention mode of CWS-MAC is then

$$S^c = \left(\frac{Q(0) - Q(k)}{t_s} \right) \frac{L_c}{R} \tag{176}$$

where L_c is the contention packet size in bits. The normalized throughput for CWS-MAC is plotted as a function of aggregate contention packet arrival rate in Figure 58 for various values of *k*. Similarly, we can use the results from Section 5 to numerically calculate the mean contention mode delay for CWS-MAC. These results are plotted in Figure 59 again as a function of the aggregate arrival rate for various values of *k*.

We have also plotted the mean residual packets remaining at the end of an active period as well as the contention mode utilization as a function of the aggregate arrival rate in Figures 60 and 61 for various values of *k*. As the utilization approaches one, the mean number of residual packets begins to rise sharply. This is an indication of saturation of the contention mode. This can clearly be seen by comparing Figures 59 and 61. As the utilization approaches one, the delay becomes unbounded.

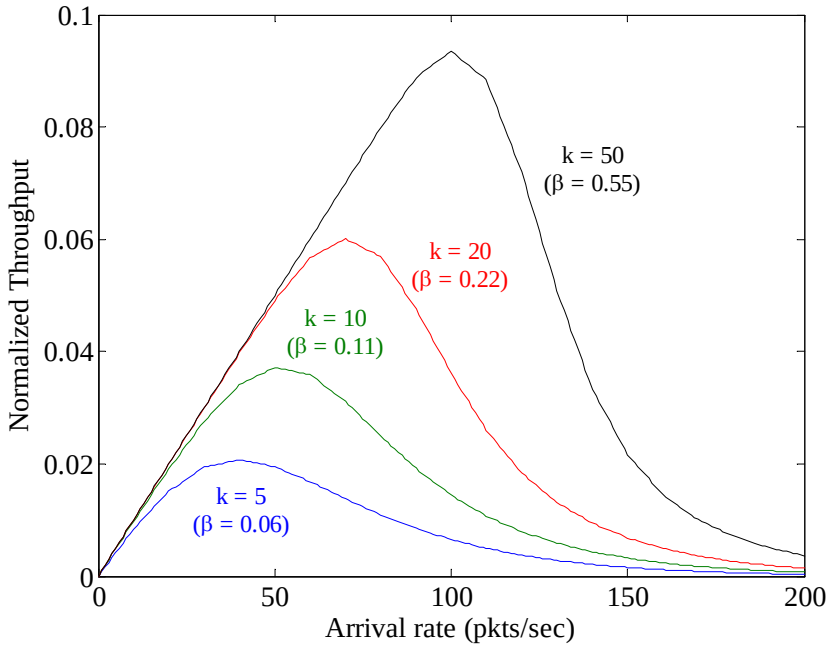


Figure 58. Normalized throughput as a function of the aggregate arrival rate for the contention mode of CWS-MAC for various values of *k*. For this plot, the probability of transmission in a slot, *p*, is 0.3.

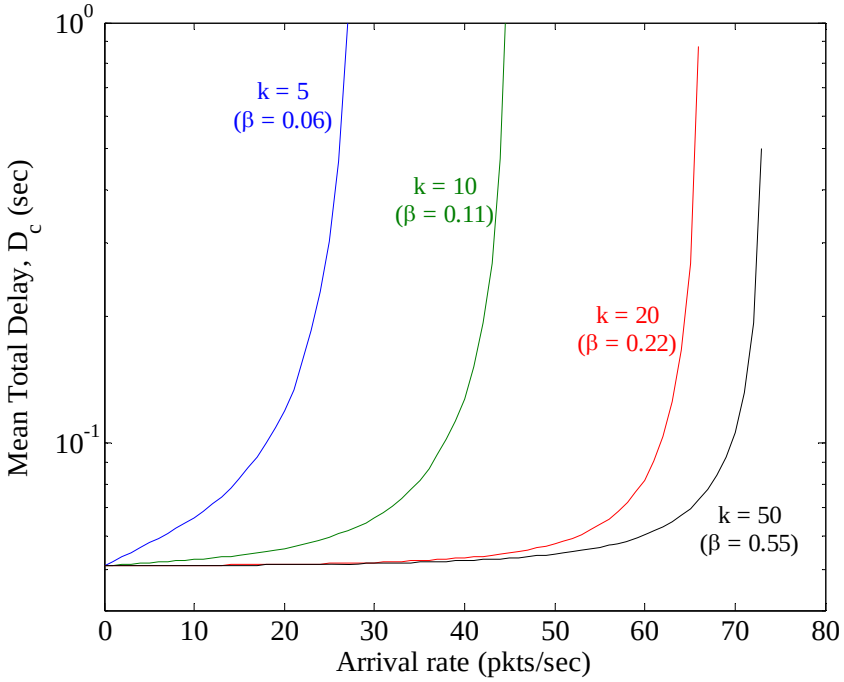


Figure 59. Mean total delay as a function of the aggregate arrival rate for the contention mode of CWS-MAC for various values of *k*. For this plot, the probability of transmission in a slot, *p*, is 0.3.

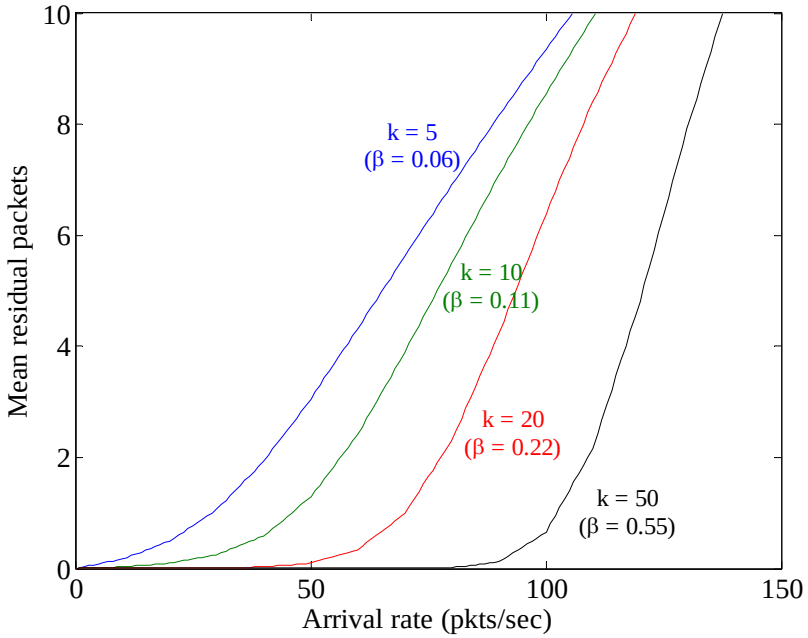


Figure 60. Mean residual packets remaining at the end of the active period as a function of the aggregate arrival rate for the contention mode of CWS-MAC for various values of *k*. For this plot, the probability of transmission in a slot, *p*, is 0.3.

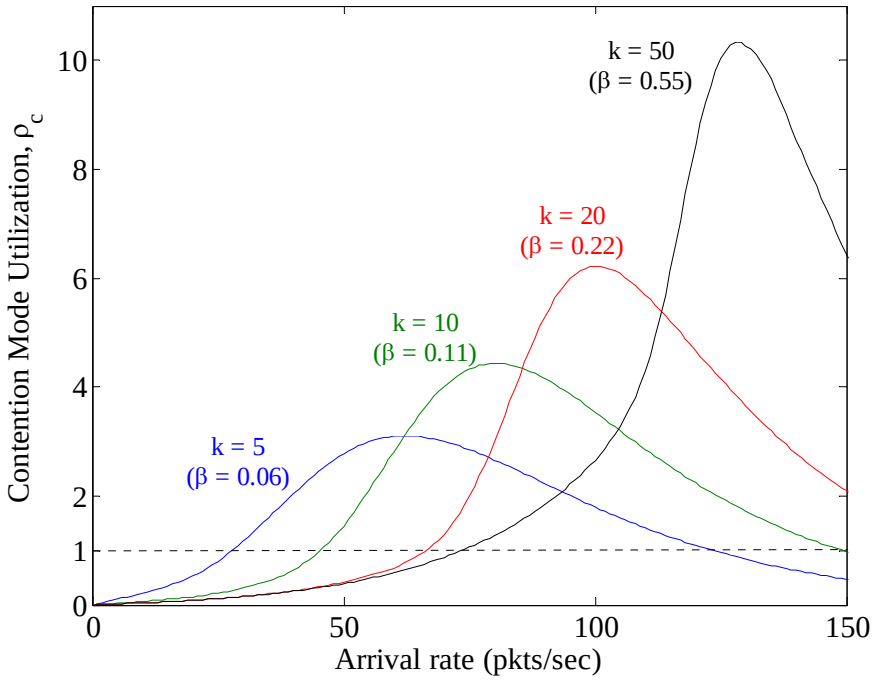


Figure 61. Utilization as a function of the aggregate arrival rate for the contention mode of CWS-MAC for various values of *k*. For this plot, the probability of transmission in a slot, *p*, is 0.3.

7. Delay and Throughput for Traffic-adaptive CWS-MAC

The results of the previous section can now be applied to the traffic-adaptive model derived in Section B for the example of Chapter III. State 1 of Figure 32 now represents the case where both flows are in the contention mode of traffic-adaptive CWS-MAC while State 2 represents the case where one flow is in the contention mode, but the other has been transitioned to the non-contention mode of traffic-adaptive CWS-MAC. Accordingly, the contention performance parameters S^c and D^c can be calculated, as in the previous section, using the combined aggregate flow arrival rate (calculated as the sum of the aggregate arrivals rates for flow 1 and flow 2). Similarly, S_1^c and D_1^c can also be calculated, as in the previous section, this time using the arrival rate of flow 1 (assumed to be the flow that is constant and remains in the contention mode). The non-contention performance parameters S_2^{nc} and D_2^{nc} can be calculated from (82) and (149), respectively. The results for the mean total delay for the example of Chapter III are

plotted in Figures 62 and 63 as function of the combined aggregate packet arrival rate for the various medium access approaches. As expected, traffic-adaptive CWS-MAC outperforms both the contention only and the non-contention only modes as well as the hybrid approach. It can also be seen that, again as expected, the performance of traffic-adaptive CWS-MAC is dependent on the effective choice of the queue-based threshold.

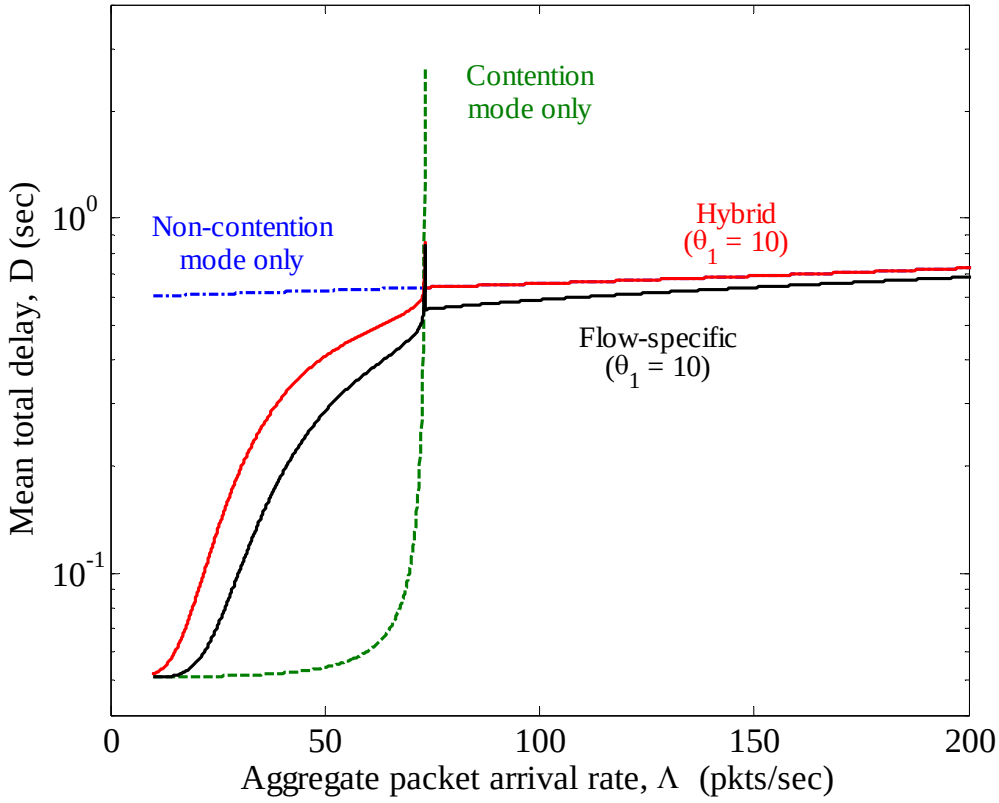


Figure 62. Mean total delay for traffic-adaptive CWS-MAC as a function of aggregate arrival rate for contention mode, non-contention mode, hybrid and flow-specific modes for the example of the previous section. For this plot, the probability of transmission in a slot, *p*, is 0.3, channel rate is 1 Mbps, packet size is 1000 bits, and there are 100 slots (one packet length in duration) in the TDMA frame.

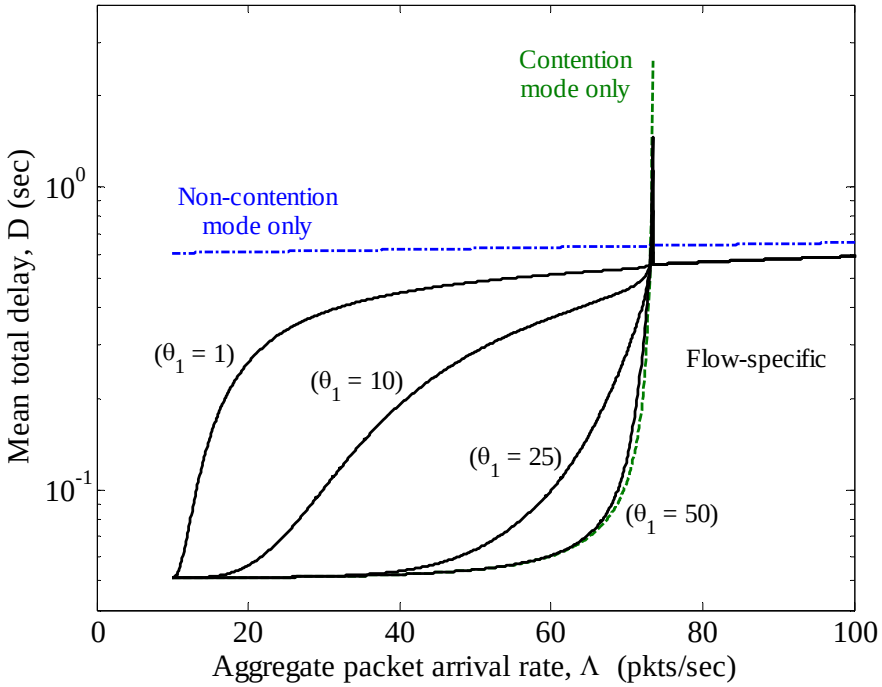


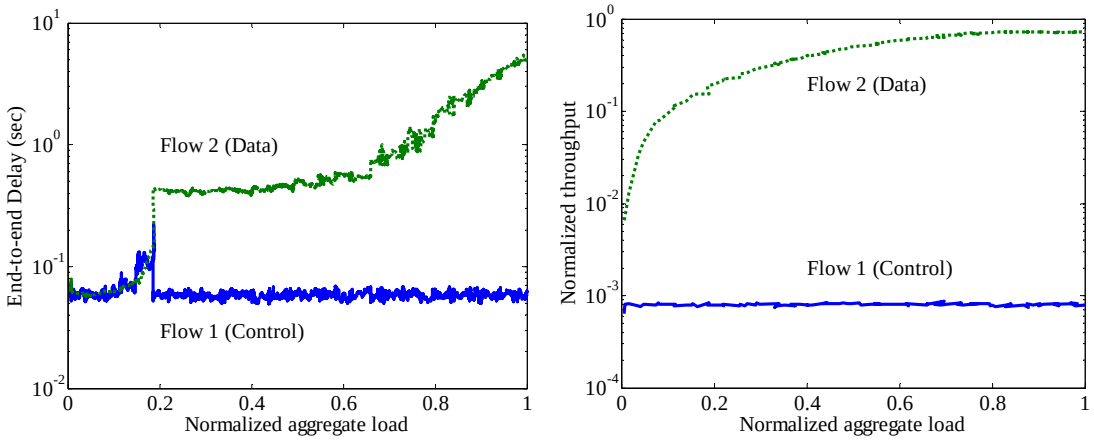
Figure 63. Mean total delay for traffic-adaptive CWS-MAC as a function of aggregate arrival rate for various values of θ_1 for flow-specific mode for the example of the previous section. For this plot, the probability of transmission in a slot, *p*, is 0.3, channel rate is 1 Mbps, packet size is 1000 bits, and there are 100 slots (one packet length in duration) in the TDMA frame.

E. SIMULATION RESULTS FOR TRAFFIC-ADAPTIVE CWS-MAC

In this section, simulation results using the OPNET® Modeler suite are provided to demonstrate the effectiveness of the traffic-adaptive, flow-specific scheme and compare it to the analysis of the previous section. For the following transient simulations, flow 1 load is kept constant at 800 bits/sec (8 packets/sec with a packet size of 100 bits). Flow 2 load is increased from zero to a maximum of the channel data rate of 1 Mbps (using a packet size of 1000 bits). Flow 1 represents the fixed rate control flow in the example of Chapter III, while flow 2 represents the variable data flow. In both cases, the packet size is constant and the packet inter-arrival times are exponentially distributed. The results were generated with a slot size of 0.1 s, a minislot size of 1 ms, a control beacon length of 1 ms, an interframe space of 0.1 ms, 50 minislots per time slot, and a neighborhood size of 8 nodes where each node is assigned a single slot. The transmission

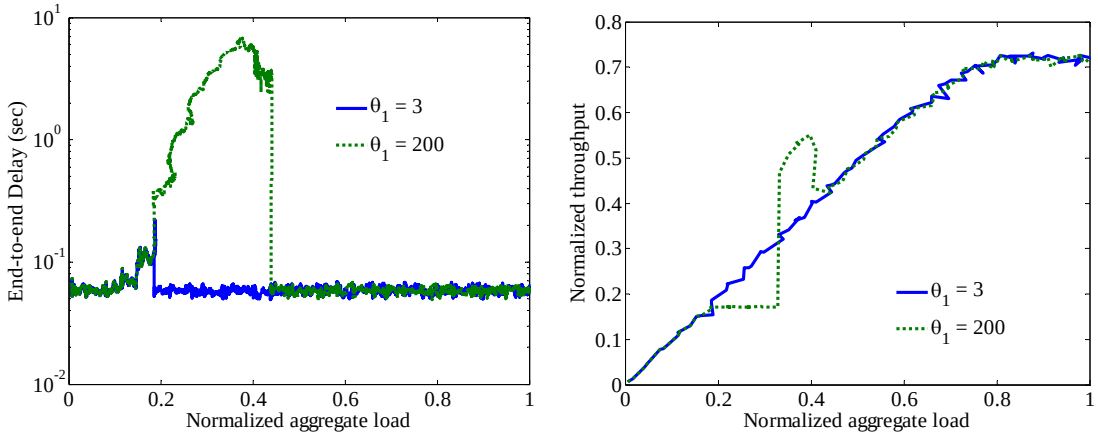
probability in each minislot was chosen as the inverse of the size of the neighborhood. The plotted results are based on Monte Carlo simulations averaged across 100 runs. Pertinent portions of the OPNET® simulation code are provided in the Appendix.

End-to-end delay and normalized throughput for both flows are presented in Figures 64(a) and 64(b), respectively. With $\theta_1 = 3$ close to optimum, it can be seen that the scheme transitions flow 2 from contention-based to contention-free access as the contention-based mode becomes saturated and the end-to-end packet delay begins to rise. This transition protects the delay bound on flow 1 while providing higher throughput for the heavy load of flow 2. In Figures 65(a) and 65(b), we can compare the performance of different values of θ_1 by taking a closer look at the delay of flow 1 and the throughput of flow 2. For the non-optimum choice of $\theta_1 = 200$, we see that the contention-based mode becomes saturated prior to transition and the flow 1 delay in Figure 65(a) rises sharply while the flow 2 throughput in Figure 65(b) levels off. Figures 66(a) and 66(b) provide a comparison of the flow-specific end-to-end delay and throughput to that of CSMA and TDMA, respectively. It can be seen that the delay of flow 1 at low loads is better than TDMA while the throughput of flow 2 at high loads is better than CSMA. The CSMA results provided represent best case delay performance as they assume head-of-the-queue privilege for flow 1 and do not include an acknowledgment mechanism.



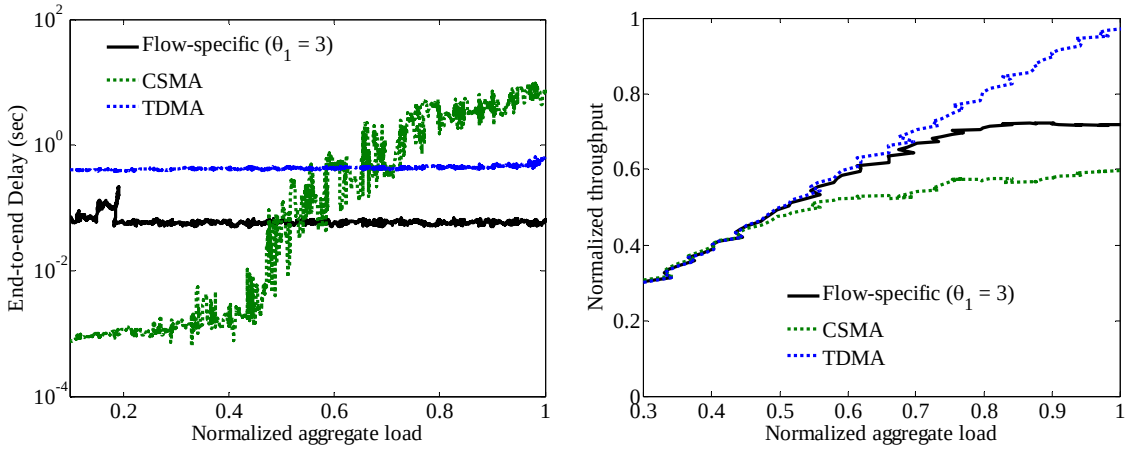
(a) (b)

Figure 64. (a) End-to-end delay and (b) normalized throughput for flow 1 (control) and flow 2 (data) plotted against normalized aggregate load $(\theta_1 = 3)$.



(a) (b)

Figure 65. (a) Flow 1 end-to-end delay plotted as a function of normalized aggregate load for $\theta_1 = 3$ and $\theta_1 = 200$. (b) Flow 2 throughput plotted as a function of normalized aggregate load for $\theta_1 = 3$ and $\theta_1 = 200$.



(a) (b)

Figure 66. (a) Flow 1 end-to-end delay and (b) flow 2 throughput compared to CSMA and TDMA for $\theta_1 = 3$.

To point out the limitations of the analysis in the previous section, the steady state mean end-to-end delay results are plotted along with the analysis results in Figure 67. In this plot, the neighborhood size is 10 nodes. In these simulations, the packet arrival rate within both of the flows is kept constant and the network is allowed to reach steady state. We then plot the results for different values of λ_2. As anticipated, the analysis does a good job of predicting performance in both the contention and non-contention regions,

but falls short in the transition region. This is not surprising and arises from the assumptions in the two-mode, two-flow analysis. By using the work of [151] and [152], we are implicitly assuming that all of the nodes transition the data flow simultaneously (or near-simultaneously). More specifically, we are assuming that the transition at one node does not affect the performance and subsequent transition at another node. In practice, of course, the transition at one node affects performance (and, ultimately, delays the transition) at all other nodes. Accordingly, the analysis of the previous section falls short in modeling these staggered transitions and their effect on each other in this transition region.

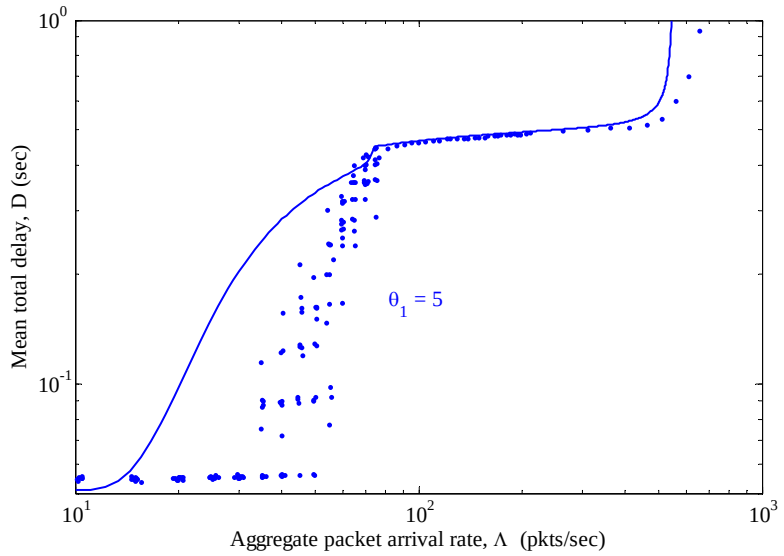


Figure 67. Comparison of the analysis in the previous section (solid line) with the steady state simulation results (discrete points on plot) for the mean end-to-end packet delay of traffic-adaptive CWS-MAC with $\theta_1 = 5$.

In this chapter, we proposed and analyzed the novel traffic-adaptive, flow-specific medium access protocol, traffic-adaptive CWS-MAC. This medium access scheme provides both per flow medium access service and a queue-based, traffic-adaptive mechanism to allow it to respond to changes in both per flow load and over network contention levels. Through both analysis and simulation, we demonstrated that it outperformed traditional contention-based, contention-free and hybrid approaches such as those based on CSMA and TDMA. We now move on to an energy-efficient implementation of traffic-adaptive CWS-MAC using preamble sampling.

V. ENERGY-EFFICIENT, FLOW-SPECIFIC MEDIUM ACCESS

Having implemented, in Chapter IV, both the per flow medium access service and the traffic-adaptive switching mechanism identified in Chapter III, we now turn our attention to the energy efficiency of the proposed solution.

As discussed in Chapter II, extending battery life in wireless sensor networks presents a fundamental energy challenge that impacts both sensor node processing and communication solutions, particularly among a group of unattended sensor nodes participating in a collaborative environment. Accordingly, we start by analyzing the energy consumption of centralized and distributed solutions in wireless sensor network applications. We look closely at the advantage of the preamble sampling technique and then apply it to develop an energy-efficient, flow-specific medium access scheme. The result is an energy-efficient version of traffic-adaptive CWS-MAC that is appropriate for implementation in the energy-constrained environment of wireless sensor networks.

This chapter begins with a comparison of the energy efficiency of centralized and distributed solutions in wireless sensor networks and develops a framework to compare the two. This work highlights the advantage of preamble sampling which is then utilized to propose an energy efficient version of traffic-adaptive CWS-MAC in the second half of the chapter. Throughput and delay performance as well as duty cycle analysis is provided and the chapter closes with relevant simulation results to validate the analysis.

A. ENERGY EFFICIENCY OF CENTRALIZED AND DISTRIBUTED SOLUTIONS IN WIRELESS SENSOR NETWORKS

The beamforming solution for unattended battlefield monitoring [4] discussed in Chapter I is a good example of a computationally complex and energy-intensive application that is difficult to implement in energy-constrained wireless sensor networks. Initial proposals have suggested a centralized, cluster-based solution [1] while follow-on work proposed energy-efficient distributed solutions [155],[156]. As we will demonstrate, these solutions do not fully account for the energy consumption of the multi-hop network and the choice of a centralized versus a distributed solution remains

an open research question for these types of military applications. In this section, we analyze the relative energy consumption of centralized and distributed solutions for multi-hop wireless sensor networks, develop an accompanying mathematical framework and apply it to the proposed beamforming solution [4] to produce a system level analysis (using Mica2, MicaZ and the latest generation Telos sensor motes [94]) of the energy trade-offs between the two approaches in unattended battlefield monitoring applications. An energy performance threshold point is found to exist between these centralized and distributed solutions, which can be exploited through the use of preamble sampling.

1. Energy Consumption of Centralized and Distributed Solutions in Wireless Sensor Networks

In this section, we derive the energy expressions for both centralized and distributed computation in multi-hop wireless sensor networks. We begin with a set of definitions and then develop generalized expressions for the centralized and distributed approaches.

A number of metrics have been proposed to capture energy consumption in a wireless sensor network [161],[162]. For the purposes of this work, we focus on the total network energy consumption and the maximum per node energy consumption. The former reflects the overall energy cost of a computation, but does not capture the fact that a single node may be disproportionately overburdened. Thus, we use the latter, per node, metric to capture this. This per node metric, when combined with current battery level information, can be used to estimate time to node failure for a given node and, when appropriate, time to network partition [161].

a. Definitions

The total energy, E_{total}, consumed in the computation effort can be classified into the energy consumed in computation, E_{comp}, and the energy consumed in communication, E_{comm}. The computation energy consumption is proportional to the number of operations required, N_{ops}, and can be expressed as

$$E_{comp} = N_{ops} R_{comp} P_{comp} \tag{177}$$

where R_{comp} is the data computation rate in operations per second and the computation power, P_{comp}, is the product of the processor current draw, i_{comp}, and the supply voltage, *V*. The energy consumption due to communication, E_{comm}, is a function of the energy consumption due to reception, E_{rx}, and the energy consumption due to transmission, E_{tx}, where

$$\begin{aligned} E_{rx} &= t_{rx} P_{rx} = \frac{L_{rx}}{R} i_{rx} V, \\ E_{tx} &= t_{tx} P_{tx} = \frac{L_{tx}}{R} i_{tx} V, \end{aligned} \tag{178}$$

t_{rx} and t_{tx} are the reception and transmission time, respectively, L_{rx} and L_{tx} are the reception and transmission packet sizes (in bits), *R* is the data rate in bits per second and *P*, *i*, and *V* are defined as in above. The superscripts *R*, *P*, and *D* will be used in conjunction with the energy terms to indicate the energy consumption associated with the raw, processed and distributed packets, respectively (e.g., E_{rx}^{R} is the energy consumption required for reception of a raw data packet).

To facilitate this analysis, we define η_{comp} as the ratio of the energy consumption for computation in the distributed approach to that in the centralized approach, which can be expressed as

$$\eta_{comp} = \frac{N_{ops}^{dist}}{N_{ops}^{cent}} \tag{179}$$

where N_{ops}^{dist} and N_{ops}^{cent} are the number of operations per node for the distributed and centralized approaches, respectively. We also define η_c as the ratio of the transmission energy to the reception energy. Assuming $L_{rx} = L_{tx}$,

$$\eta_c = \frac{i_{tx}}{i_{rx}}. \tag{180}$$

To capture the energy associated with preamble sampling, we define

$$E_{ps} = t_{ps} P_{rx} \text{ and } \eta_{ps} = \frac{E_{ps}}{E_{rx}} = \frac{L_{ps}}{L_{rx}} \tag{181}$$

where t_{ps} is the time spent receiving the preamble and L_{ps} is the size of the preamble in bits. The case when $\eta_{ps} = 1$ equates to the situation when preamble sampling is not implemented. Finally, we define

$$\eta_a = \frac{L_P}{L_R} \text{ and } \eta_d = \frac{L_D}{L_R} \tag{182}$$

to capture the relationship between the size (in bits) of a raw data packet, L_R, and the size of a processed packet in both the centralized and distributed schemes, L_P and L_D, respectively. A summary of the terms used in this section is included in Table 6.

$E_{total}^{cent}\ \left(E_{total}^{dist}\right)$	Total energy consumption, centralized (distributed) approach
$E_{comp}^{cent}\ \left(E_{comp}^{dist}\right)$	Energy consumption due to computation, centralized (distributed) approach
$E_{comm}^{cent}\ \left(E_{comm}^{dist}\right)$	Energy consumption due to communication, centralized (distributed) approach
$E_{\max}^{cent}\ \left(E_{\max}^{dist}\right)$	Maximum per node energy consumption, centralized (distributed) approach
$N_{ops}^{cent}\ \left(N_{ops}^{dist}\right)$	Number of operations required in centralized (distributed) approach
$E_{tx}^{R}, E_{tx}^{P}, E_{tx}^{D}$	Energy required for transmission of raw, processed, and distributed packets, respectively
$E_{rx}^{R}, E_{rx}^{P}, E_{rx}^{D}$	Energy required for reception of raw, processed, and distributed packets, respectively
R_{comp}, P_{comp}	Data computation rate and computational power consumption
η_{ps}	Preamble sampling ratio
η_c	Ratio of transmission energy consumption to reception energy consumption
$\eta_a\ (\eta_d)$	Ratio of processed to raw packet size in centralized (distributed) approach
η_{comp}	Ratio of number of operations required for the distributed approach to the number required for the centralized approach
D	Mean node degree (i.e., number of neighbors)
H	Mean number of hops to central controller in centralized approach
K	Mean number of iterations to convergence in distributed approach

Table 6. Terms used in the development of the energy efficiency of centralized and distributed solutions in wireless sensor networks.

b. Energy Consumption in a Centralized Approach

In a centralized implementation, all nodes forward raw data to a central node where the computation is accomplished and the results are then transmitted back to the network nodes. The total communication energy consumption for a centralized computation approach can be divided into three parts. The first is the energy required to transmit and forward the raw packets to the central node. The second is the energy required to transmit and forward the processed packets back to the network nodes. The final component captures the energy consumption due to overhearing (i.e., the energy associated with the preamble sampling required at nodes that are not the destination node for a particular packet). For a network comprised of *N* nodes with mean degree of *D* and a mean hop count of *H* to the central node, the energy consumption due to communication for a centralized approach can be expressed as [186]

$$\begin{aligned} E_{comm}^{cent} &= H\left(N-1\right)E_{tx}^{R} + H\left(N-1\right)E_{rx}^{R} \\ &\quad + H(N-1)E_{tx}^{P} + H\left(N-1\right)E_{rx}^{P} \\ &\quad + 2H\left(N-1\right)\left(D-1\right)E_{ps}, \end{aligned} \tag{183}$$

which, after making the appropriate substitutions and rearranging the terms, is equivalent to

$$\begin{aligned} E_{comm}^{cent} &= H\left(N-1\right)\left[\left(1+\eta_c+\eta_a\eta_c+\eta_a\right)\right. \\ &\quad \left. +2\left(D-1\right)\eta_{ps}\right]E_{rx}^{R}. \end{aligned} \tag{184}$$

The resulting total energy consumption for the network is then

$$\begin{aligned} E_{total}^{cent} &= E_{comm}^{cent} + E_{comp}^{cent} \\ &= H\left(N-1\right)\left[\left(1+\eta_c+\eta_a\eta_c+\eta_a\right)\right. \\ &\quad \left. +2\left(D-1\right)\eta_{ps}\right]E_{rx}^{R} + N_{ops}^{cent}R_{comp}P_{comp}. \end{aligned} \tag{185}$$

In Figure 68, we plot the relative communication energy consumption (normalized by E_{rx}^{R}) as a function of the number of nodes in the network for various values of mean hop count and η_{ps}. While the energy consumption increases linearly with the mean hop count as expected, what is interesting is the effect of η_{ps}. The limiting case

where $\eta_{ps} \to 0$ shows more than an order of magnitude improvement over the case when preamble sampling is not employed ($\eta_{ps} = 1$).

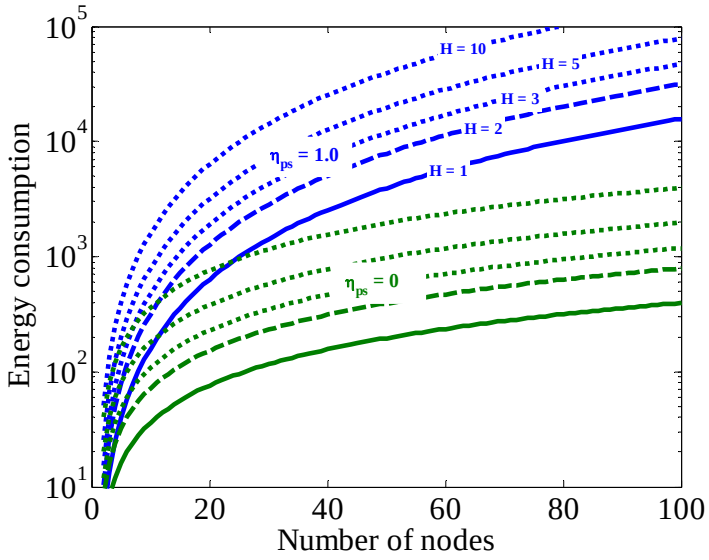


Figure 68. Relative energy consumption (normalized by E_{rx}^{R}) of the centralized approach plotted as a function of the number of nodes in the network for multiple mean hop count values with $\eta_c = \eta_a = 1$.

We can gain insight into the effect of the communication range on the energy consumption of the centralized solution by utilizing the findings of [163] and [164] to relate communication range to mean number of neighbors (node degree) and the mean hop count. Assuming that the nodes are distributed homogenously over the 2-D plane according to a Poisson process with intensity λ, the node degree can be shown to be [164]

$$D = \lambda \pi r^2 e^{2\left(\frac{h\sigma}{\delta}\right)^2} \tag{186}$$

where $h = \frac{\ln 10}{10}$, δ is the path loss exponent, and the fading is assumed to be log-normally distributed with mean of zero and variance of σ^2. A minimum bound for the mean hop count can be arrived at if we ignore the effect of fading (set $\sigma = 0$) and assume that there is always a relay node at distance *r* in the direction of the destination node. For a rectangular area of size $a \times b$, this bound can be shown to be [163]

$$H \geq \frac{1}{15r}\left[\frac{a^3}{b^2}+\frac{b^3}{a^2}+\sqrt{a^2+b^2}\left(3-\frac{a^2}{b^2}-\frac{b^2}{a^2}\right)\right] + \frac{1}{6r}\left[\frac{b^2}{a}\operatorname{arcosh}\frac{\sqrt{a^2+b^2}}{b} + \frac{a^2}{b}\operatorname{arcosh}\frac{\sqrt{a^2+b^2}}{a}\right] \quad (187)$$

where $\operatorname{arcosh}(x) = \ln\left(x+\sqrt{x^2-1}\right)$. For a square area of size $a \times a$, this reduces to $H \geq 0.5124\left(\frac{a}{r}\right)$. Using these results, we plot the energy consumption as a function of communication range for several values of η_{ps} in Figure 69. Again, it can be seen that preamble sampling significantly reduces energy consumption for the centralized approach. For the limiting case where $\eta_{ps} \to 0$, energy consumption actually decreases as a function of increasing communication range. This occurs because no energy is spent receiving packets for which a node is not the intended destination in this limiting case.

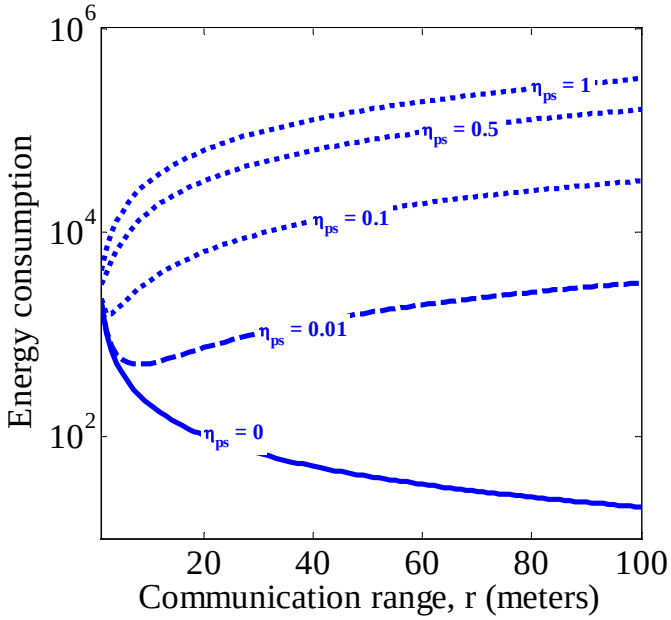


Figure 69. Relative energy consumption (normalized by E_{rx}^{R}) of the centralized approach plotted as a function of communication range for multiple values of η_{ps}.

We can provide a lower bound for the maximum per node energy consumption of the centralized approach by assuming it occurs at the central node. This is a lower bound because energy consumption due to forwarding may result in higher energy consumption at an intermediate node (which is topology specific). This lower bound is comprised of the computation energy and the energy required to receive the raw data packets and transmit the processed data packets. Thus, the maximum per node energy for the centralized approach is bounded as

$$E_{\max}^{cent} \geq \left(N-1\right)E_{rx}^{R} + \left(N-1\right)E_{tx}^{P} + N_{ops}^{cent}R_{comp}P_{comp}\,, \tag{188}$$

which is equivalent to

$$E_{\max}^{cent} \geq \left(N-1\right)\left[1+\eta_c\eta_a\right]E_{rx}^{R} + N_{ops}^{cent}R_{comp}P_{comp}\,. \tag{189}$$

c. Energy Consumption in a Distributed Approach

In a distributed approach, nodes share the computational burden through local processing and information exchange and conduct a series of iterations to converge to a global solution. The energy required for communication in the distributed approach is the energy required in each iteration to transmit and receive the locally processed packets within each one-hop neighborhood. For *K* iterations in a network of *N* nodes with mean degree *D*, the energy consumption due to communication for a distributed approach is

$$E_{comm}^{dist} = KN\left(E_{tx}^{D} + DE_{rx}^{D}\right). \tag{190}$$

Again, after rearranging terms and making the appropriate substitutions, this is equivalent to

$$E_{comm}^{dist} = KN\eta_d\left(\eta_c + D\right)E_{rx}^{R}. \tag{191}$$

The total energy for this approach is then

$$\begin{aligned} E_{total}^{dist} &= E_{comm}^{dist} + KNE_{comp}^{dist} \\ &= KN\eta_d\left(\eta_c + D\right)E_{rx}^{R} + KN\eta_{comp}E_{comp}^{cent}. \end{aligned} \tag{192}$$

As the computation and communication load is spread out among all network nodes in the distributed approach, the maximum per node energy is nominally the same at all nodes and is given by

$$E_{\max}^{dist} = K\left(E_{tx}^{D} + DE_{rx}^{D}\right) + N_{ops}^{dist} R_{comp} P_{comp}\,, \tag{193}$$

which is equivalent to

$$E_{\max}^{dist} = K\eta_d\left(\eta_c + D\right)E_{rx}^{R} + \eta_{comp} E_{comp}^{cent}\,. \tag{194}$$

In Figures 70 and 71, we plot the relative energy consumption due to communication for the distributed approach with multiple iterations as a function of the number of nodes and the communication range, respectively. As expected, the energy consumption grows linearly with both.

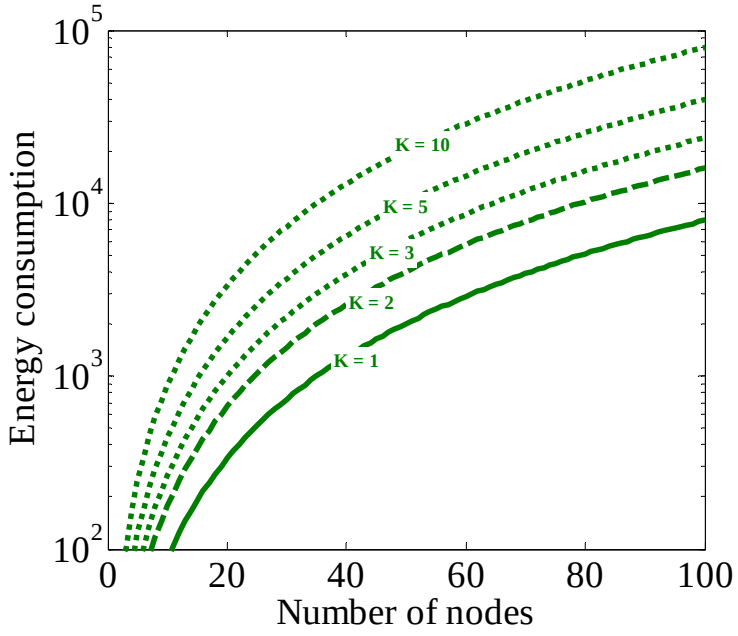


Figure 70. Relative energy consumption (normalized by E_{rx}^{R}) of the distributed approach plotted as a function of the number of nodes in the network for multiple iteration values.

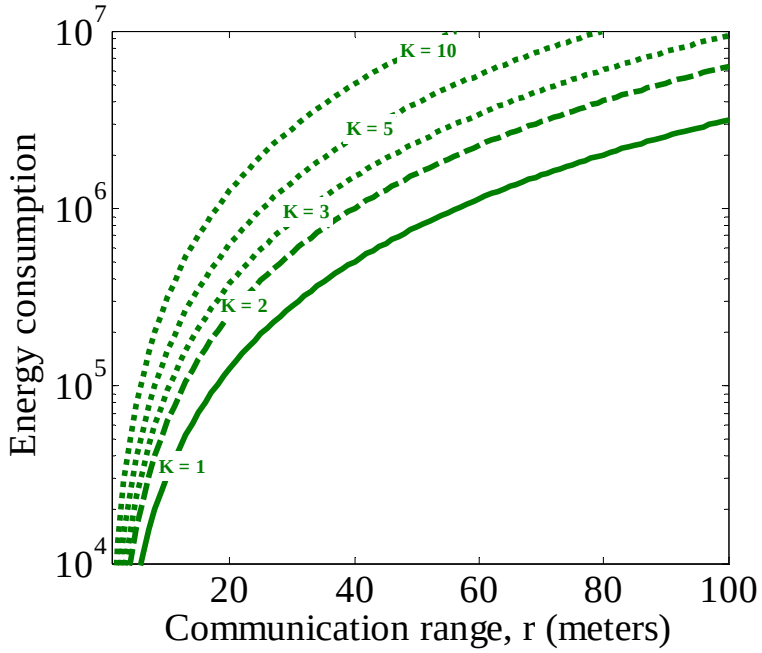


Figure 71. Relative energy consumption (normalized by E_{rx}^{R}) of the distributed approach plotted as a function of the communication range for multiple iteration values.

2. A Comparison of the Energy Efficiency of Centralized and Distributed Approaches

We compare the energy efficiency performance of the centralized and distributed approaches using the results of the previous section. For this work, we focus on the relative energy consumption due to communication, which has been shown to dominate the total energy consumption [17]. As in the previous sections, we normalize the energy consumption by the energy consumption due to reception of a raw data packet, E_{rx}^{R}.

In Figure 72, we plot the energy consumption as a function of the number of nodes for multiple values of the hop count to the central controller in the centralized case and number of iterations required for convergence in the distributed case. It can be seen that when preamble sampling is not utilized ($\eta_{ps} = 1$), the distributed approach outperforms the centralized approach. However, in the limiting case where $\eta_{ps} \to 0$, the centralized approach will outperform the distributed approach for reasonable values of hop count to the central controller.

To examine this result closer, we plot the energy consumption as a function of η_{ps} in Figure 73. The energy consumption of the distributed approach is not a function of η_{ps} and appears as a constant in the plot, while the performance of the centralized approach, on the other hand, varies dramatically with the implementation of preamble sampling. This is because the broadcast transmissions in the distributed approach are targeted at all nodes within communication range of the transmitter while the transmissions in the centralized approach are only intended for the next hop node in the routing paths to and from the central controller. It can be seen in Figure 73 that preamble sampling significantly improves the energy efficiency of the centralized algorithms and there exists a threshold below which the centralized approach will outperform the distributed approach. This threshold is a function of the number of hops to the central controller in the centralized case and the number of iterations required for convergence in the distributed case.

3. Energy Efficiency of Centralized and Distributed Beamforming Solutions for Unattended Battlefield Monitoring

We now return to the beamforming solution for unattended battlefield monitoring and analyze the relative energy efficiency of the distributed and centralized algorithms of [155] and [156]. In this analysis, we demonstrate the efficiency gains that can be achieved by incorporating preamble sampling. To support the weight calculations required in beamforming, [155] and [156] propose distributed solutions to the least squares problem in which the QR factorization of the steering matrix is accomplished through the use of Householder transformations [165],[166].

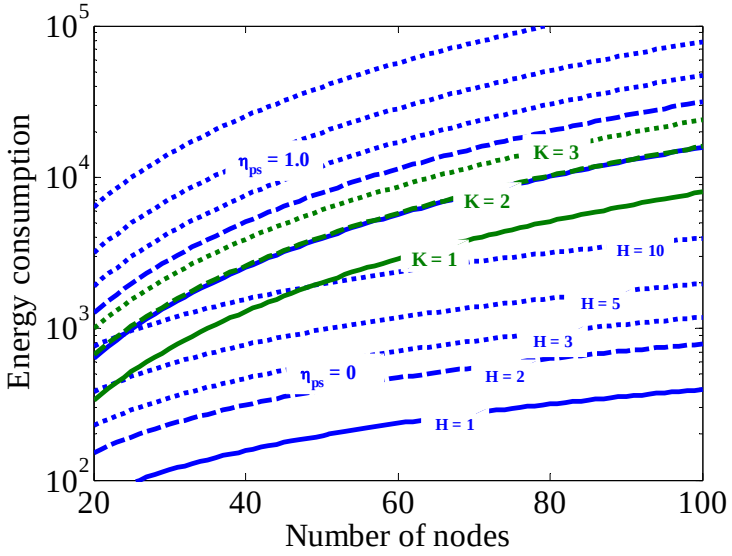


Figure 72. A comparison of relative energy consumption (normalized by E_{rx}^{R}) of the centralized (blue) and distributed (green) approaches plotted as a function of the number of nodes in the network for various mean hop count values and iteration values with $\eta_c = \eta_a = 1$.

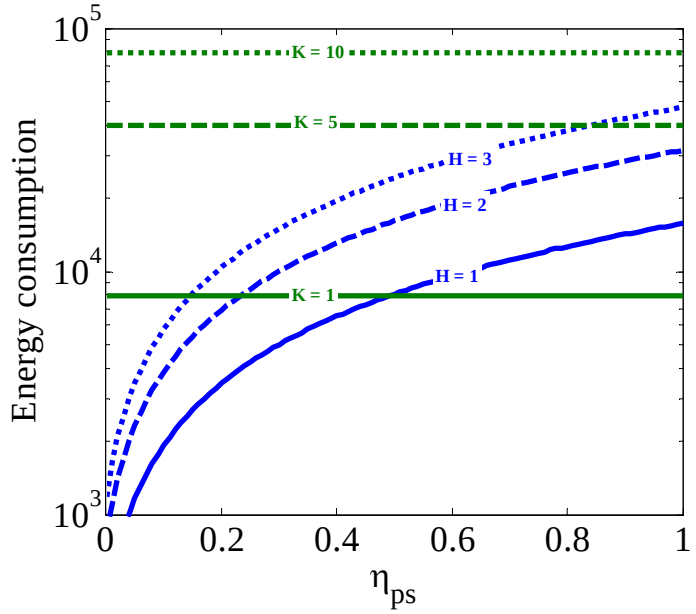


Figure 73. A comparison of relative energy consumption (normalized by E_{rx}^{R}) of the centralized (blue) and distributed (green) approaches plotted as a function of η_{ps} for various mean hop count values and iteration values with $\eta_c = \eta_a = 1$.

In the baseline centralized approach of [1] (labeled “centralized” in the following plots), all nodes transmit their location information to the central controller, which then calculates the weights and returns them to the nodes. The total number of operations performed by the central controller can be shown to be [165]

$$N_{ops}^{cent} = 2N^2\left(m - \frac{N}{3}\right) + mN + N^2 \tag{195}$$

where m is the number of rows in the beamforming steering matrix and N is the number of nodes. We can substitute (195) into (185) and (189) to calculate the energy consumption of this centralized approach.

In the distributed approach of [155] (labeled “distributed” in the following plots), the columns of the steering matrix are distributed among the nodes and the QR factorization is performed for each column locally at the appropriate node and shared among all nodes. The weights are then calculated through back substitution and the results for each node are again broadcast to the other nodes. It can be shown that the total number of operations remain the same as in the centralized solution of (195) [155]. The total number of messages transmitted in this approach is [155]

$$M^{dist1} = \left(m + 4 - \frac{N}{2}\right)(N-1). \tag{196}$$

We will make use of (192) and (194) to calculate the energy consumption of this distributed approach. Although there is only one iteration identified in the published algorithm, the number of messages in (196) must be accounted for by recognizing that these messages, in effect, constitute one iteration while a second iteration is required to then transmit the locally calculated solution. For this first iteration, we replace N with M^{dist1} in (192).

In the distributed approach of [156] (labeled “distributive, iterative” in the following plots), the columns are again distributed among the nodes, but the weight calculation is done in an iterative fashion. The total number of operations are shown to be [156]

$$N_{ops}^{dist2} = N\left[2\left(m - \frac{1}{3}\right) + k(3m+1)\right] \tag{197}$$

where k is the number of iterations required to reach convergence. In [156], it is suggested that a nominal value for k is three. We again substitute these into (192) and (194) to calculate the energy consumption of this distributed, iterative approach.

Using the framework outlined in the previous sections and the sensor node operating parameters in Table 7, we now compare the energy efficiency of the two distributed approaches to that of the centralized solution for implementations using Mica2, MicaZ, and Telos sensor motes [94]. For the purposes of this analysis, we assume that the centralized algorithm produces one processed packet that is the same size as the sum of the raw data packets from each of the nodes ($\eta_a = N - 1$) and that the packets in the distributed algorithms are the same size as the raw data packets in the centralized approach ($\eta_d = 1$).

	Mica2	**MicaZ**	**Telos**
i_{rx}	15.1 mA	23.3 mA	21.8 mA
i_{tx} (0 dBm)	25.4 mA	21.0 mA	19.5 mA
η_c	1.682	0.901	0.894

Table 7. Operating parameters for Mica2, MicaZ, and Telos sensor motes (After [94])

Focusing our attention on the energy consumption due to communication, we provide a comparison of the energy efficiency of the three approaches as a function of the number of nodes in Figure 74. The distributed, iterative approach of [156] can be seen to outperform both the distributed approach of [155] and the centralized approach for the majority of multi-hop cases. Of interest, though, is that the centralized approach is more efficient in terms of total energy consumption than the distributed approach. This highlights the trade-off between total energy consumption and per node energy consumption. Here, the distributed algorithm results in larger overall energy consumption in its attempt to distribute the computational load and realize smaller per node consumption. In contrast, the distributive, iterative approach achieves both reduced overall energy consumption and reduced per node energy consumption. As expected, the energy consumption rises linearly with the increasing number of nodes.

The effect of implementations with different sensor motes is shown in Figure 75 where we compare the energy efficiency as a function of the number of nodes for both Mica2 and Telos implementations. The newer generation Telos sensor motes are seen to only slightly outperform the older generation Mica2 motes. This is because the relative performance improvement is impacted by the increased reception energy consumption of the Telos mote.

Finally, in Figure 76, we examine the impact of the use of preamble sampling in the proposed solutions and see that, for effective preamble sampling schemes where η_{ps} is low, the centralized approach is capable of outperforming even the distributed, iterative approach. This is an important finding and suggests the implementation of centralized computational algorithms coupled with preamble sampling energy efficiency techniques to reduce overall energy consumption in the wireless sensor network. A clustering approach, such as that proposed in [1] can be utilized in conjunction with the centralized approach to distribute the processing load among the member nodes as desired. We apply preamble sampling to traffic-adaptive CWS-MAC in the following section to arrive at an energy-efficient, traffic-adaptive, flow-specific solution that is suitable for implementation in wireless sensor networks.

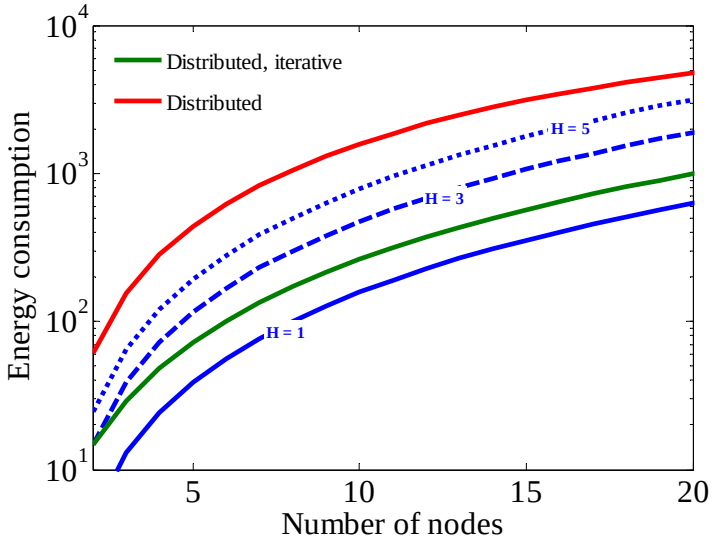


Figure 74. Relative communication energy consumption (normalized by E_{rx}^{R}) for the centralized and distributed approaches of [155] and [156] as a function of the number of nodes for the Telos sensor motes.

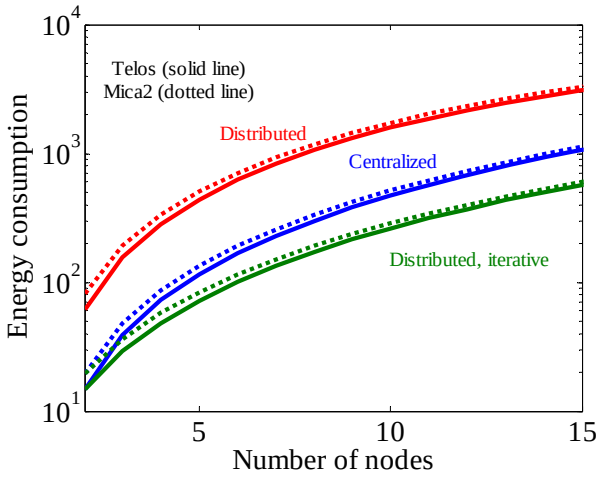


Figure 75. Relative communication energy consumption (normalized by E_{rx}^{R}) for the centralized (with $H = 3$) and distributed approaches of [155] and [156] as a function of the number of nodes for Mica2 and Telos sensor motes.

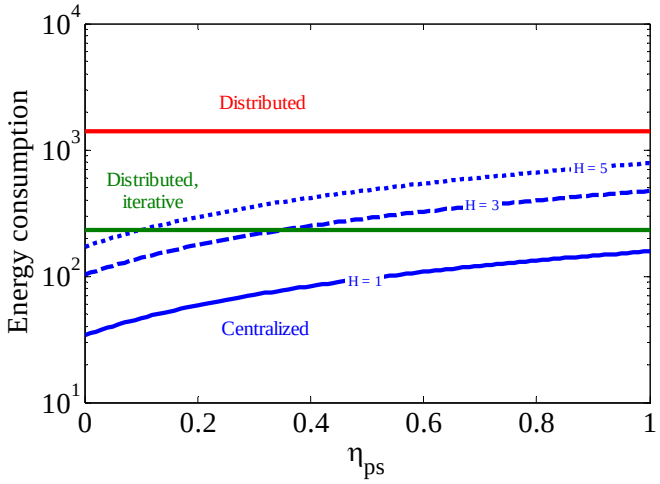


Figure 76. The effect of preamble sampling on communication energy consumption for the centralized and distributed approaches of [155] and [156]. Relative communication energy consumption (normalized by E_{rx}^{R}) for a network of 10 Telos motes is plotted as a function of η_{ps}.

B. ENERGY-EFFICIENT, FLOW-SPECIFIC MEDIUM ACCESS USING PREAMBLE SAMPLING

Building on the results of the previous section, the major contribution of the work covered in this chapter is an energy-efficient, flow-specific medium access scheme that is capable of achieving the low duty cycles required in current power-constrained wireless

sensor applications while providing the throughput and delay performance advantages of a flow-specific approach. We accomplish this by incorporating an adaptive sleep and wake cycle into the traffic-adaptive CWS-MAC protocol using preamble sampling. This adaptive sleep and wake cycle responds to changes in traffic load and we introduce a preamble sampling probability parameter that is capable of managing the trade-off between energy efficiency and throughput and delay performance.

In this work, we use preamble sampling to minimize overhearing, idle listening, and overemitting. Collisions are reduced by switching from a contention-based medium access approach to a scheduled approach as the per flow load increases. We minimize control packet overhead by proposing a distributed scheme in which nodes are capable of making local medium access decisions without the requirement for a centralized controller. A discussion of related work in energy-efficient wireless medium access can be found in Chapter II.

1. Proposed Energy-efficient, Flow-specific Medium Access Scheme

The proposed energy-efficient flow-specific medium access control scheme is based on the traffic-adaptive CWS-MAC protocol of the previous chapter. We modify this scheme by adding a sleep state during which sensor nodes power down their transceiver and processor to reduce energy consumption. Individual nodes participate in a sleep and wake cycle that is coordinated through the use of preamble sampling. This allows individual nodes to sleep when they are not designated receiver(s) during the current transmission. In this section, we present the frame structure and discuss the operation of the proposed scheme.

a. Frame Structure

The frame structure of the proposed energy-efficient, traffic-adaptive CWS-MAC protocol is shown in Figure 77. The non-contention and contention modes of this flow-specific protocol are implemented as in Chapter IV. A frame destination bitmap is added at the beginning of each frame as shown in Figure 78. This map is used to identify the designated receivers in each transmission slot. This destination bitmap is subdivided into slots, which correspond to the slots assigned in the transmission frame. Each slot in the destination bitmap is further subdivided into another level of slots (again

corresponding to the slot structure of the transmission frame), which represent the individual slot transmission maps. The first level corresponds to the transmission slot while the second level identifies the designated receiver(s) in that transmission slot. The frame destination bitmap is of duration

$$t_{mf} \geq \frac{n^2}{R} \tag{198}$$

where n is the number of slots in the transmission frame and R is the channel data rate. This is a lower bound because the size may be increased to provide both error correction to protect the destination map field and guard bands based on the fidelity of the slot synchronization scheme utilized.

A minislot destination bitmap is also included at the beginning of each minislot within the contention slot as shown in Figure 79. This map is used to designate the receiver(s) for the subsequent contention packet transmission in that minislot. Again using a bitmap approach, it is comprised of a series of bits corresponding to each slot in the transmission frame. These bits are used to designate the intended receiver(s). The minislot destination bitmap is of duration

$$t_{mms} \geq \frac{n}{R}. \tag{199}$$

While guard bands are typically not needed in this instance, (199) remains a lower bound due to the potential use of an error correction scheme.

The destination bitmap approach takes advantage of the existing slot structure used to support the non-contention mode of the protocol. The slot assignment process can be either distributed or centralized and the results must be disseminated to all nodes identified within the transmission frame.

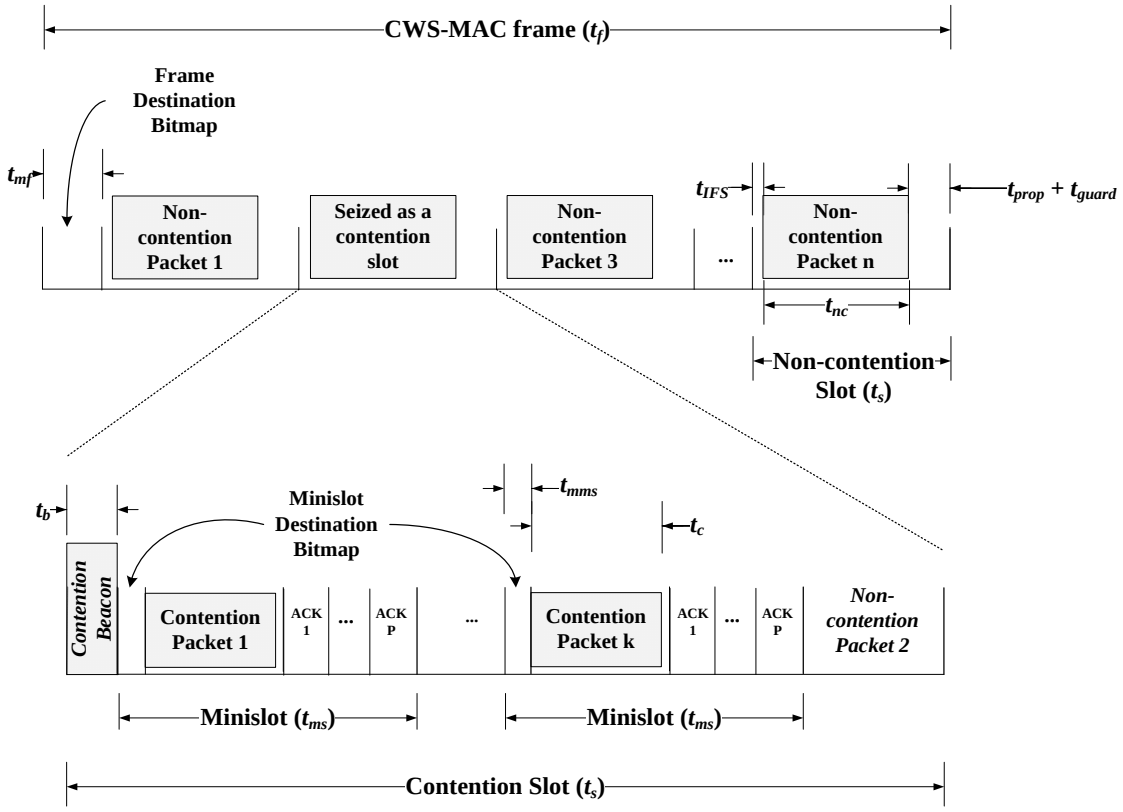


Figure 77. Frame structure of the proposed energy-efficient, traffic-adaptive CWS-MAC protocol.

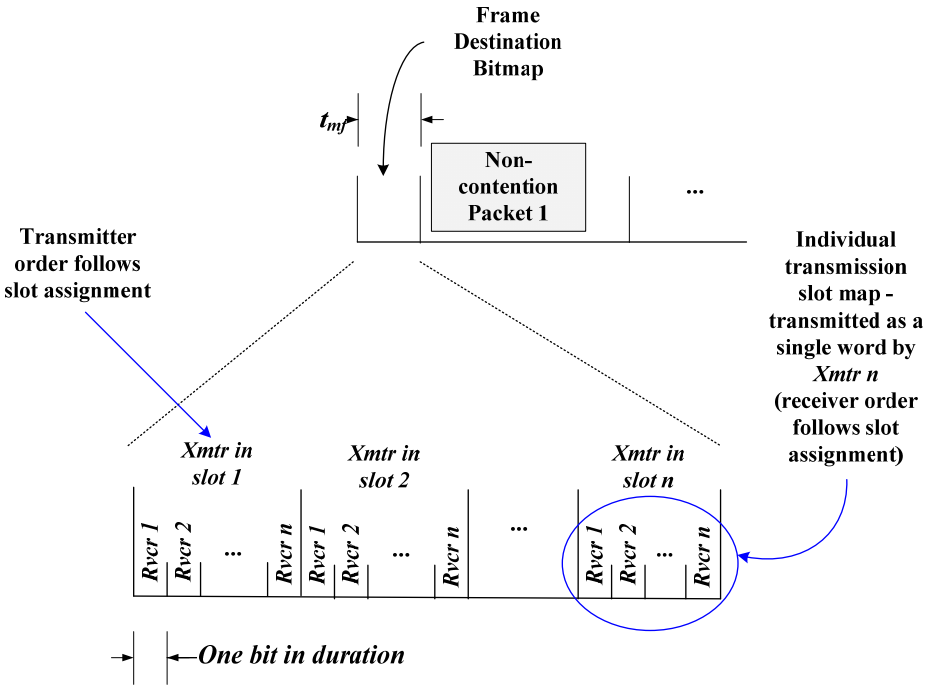


Figure 78. Frame destination bitmap.

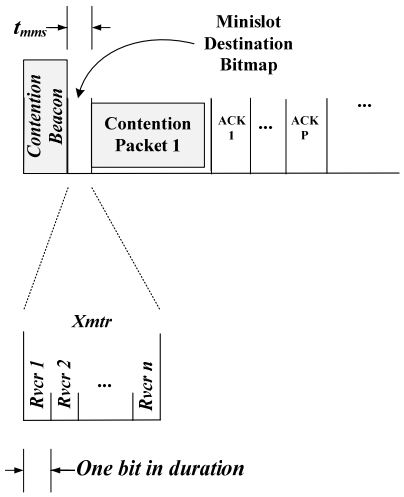


Figure 79. Minislot destination bitmap.

b. Operation

In this section, we discuss the operation of the proposed energy-efficient scheme. We begin with the frame destination bitmap and then outline operation in both the non-contention and contention modes. Nodes wake up at the beginning of each transmission frame to receive the frame destination map. This map is created by the individual nodes who broadcast their transmission slot maps at the appropriate time within the frame destination bitmap. For a given node, this individual slot map identifies the receiver(s) for the upcoming transmission in the node's slot within the current transmission frame. A bit value of one indicates that a specific node is an intended receiver for the transmission while a bit value of zero indicates that it is not. A value of all zeroes in the slot map indicates that the owner of that slot does not have any non-contention traffic to transmit. The order of the bits corresponds to the slot assignment within the transmission frame. If a node is assigned to transmit in slot *j* within the frame, then bit *j* within the slot map is used to identify it as a potential receiver in that transmission slot. Note that a non-contention packet must arrive prior to the frame to be included in the frame destination bitmap and, hence, be eligible for transmission in the current frame.

Once the frame destination map has been received and processed, nodes will know in which transmission slots they are designated as an intended receiver. In the non-contention mode, nodes enter the sleep state for the duration of transmission slots in which they are neither an intended receiver nor the slot owner with traffic to transmit.

To accommodate the contention mode, nodes will wake up with some probability p_s (denoted as the preamble sampling parameter) to sample the channel in each transmission slot at time t_{IFS} (relative to the slot boundary) for the presence of a contention beacon. When a contention beacon is detected, the slot is redesignated by the node as a contention slot. A value of $p_s = 1$ represents the case where the node samples every transmission slot. A value of $p_s = 0$ indicates that a node will only detect contention slots that occur when the node is either an intended receiver of the scheduled non-contention packet transmission or it is the slot owner and has non-contention traffic to transmit.

During a contention slot, a node will wakeup at the beginning of each minislot to receive the minislot destination bitmap to determine whether or not it is an intended receiver for the subsequent contention traffic transmission in that minislot. This minislot destination bitmap is broadcast by a node that intends to attempt transmission in that minislot. A node will stay awake for the subsequent contention packet transmission if it is designated as an intended receiver. A node is allowed to enter the sleep state until the next minislot if it is not the intended receiver or if the bitmap is unreadable as a result of a collision due to multiple attempted transmissions in the minislot. Upon termination of the final minislot, a node will remain in the sleep state unless it was the designated receiver (or transmitter) of non-contention packets in the original non-contention slot that was "seized." If this is the case, the node will wake up to receive (or transmit) non-contention packets during the non-contention period reserved at the end of the contention slot.

The use of the frame destination bitmap ensures that a node will always be awake for the transmission of a non-contention packet for which it is the intended receiver. Similarly, the use of the minislot destination bitmap ensures that a node will

always be awake for the transmission of a contention packet for which it is the intended receiver, provided that it has detected the contention slot. The probability that a node detects a contention slot is a function of the probability p_s that it will sample for the contention beacon. If a node fails to detect a contention slot, the node will sleep through the slot and will not receive any contention packets transmitted during that slot for which it is the intended receiver. The same acknowledgement mechanism that recovers from collisions within the contention slot can also be used to recover from packet losses due to missed contention slots.

2. Performance Analysis

In this section, we analyze the duty cycle performance and the throughput and delay of the proposed energy-efficient, flow-specific scheme. We also examine the effect of the sampling probability p_s on the performance of the protocol and its role in managing the tradeoff between energy consumption and throughput and delay performance. For the analysis of this section, we assume packet arrivals are Poisson-distributed and, without a loss of generality, that each node is assigned a single slot in the transmission frame.

a. Duty Cycle Analysis

We begin our duty cycle analysis by defining the duty cycle, Π, as the ratio of the mean time a node spends in the wake state during a frame, t_{awake}, to the frame time as in

$$\Pi = \frac{t_{awake}}{t_f}. \tag{200}$$

The higher the duty cycle, the more time a node stays awake and, consequently, the greater the energy consumption.

At a minimum, a node will wake up to receive the frame destination bitmap and, hence, the minimum achievable duty cycle is

$$\Pi_{\min} = \frac{t_{mf}}{t_f}. \tag{201}$$

In general, the mean duty cycle is a function of the probability that a node will wake up for a slot, the amount of time it will be awake in the slot, t_w, and the number of slots *n* in a transmission frame as in

$$\overline{\Pi} = \frac{t_{mf} + nt_w \times \Pr[\text{node will be awake for slot}]}{t_f}. \tag{202}$$

A slot can be either a non-contention slot or a contention slot, so we can expand (202) to

$$\begin{aligned} \overline{\Pi} = \frac{t_{mf}}{t_f} + n\Bigg\{ & \frac{t_{wnc}}{t_f} \times \Pr[\text{non-cont slot}] \times \\ & \Pr[\text{awake for non-cont slot}] \\ & + \frac{t_{wc}}{t_f} \times \Pr[\text{cont slot}] \times \Pr[\text{awake for cont slot}] \Bigg\} \end{aligned} \tag{203}$$

where t_{wnc} and t_{wc} are the amount of time a node will be awake in a non-contention and contention slot, respectively. From the analysis provided in Chapter IV,

$$\begin{aligned} &\Pr[\text{non-cont slot with } p_s = 1] \equiv p_0 = e^{-\Lambda_c t_s} \text{ and} \\ &\Pr[\text{cont slot with } p_s = 1] = 1 - p_0 = 1 - e^{-\Lambda_c t_s} \end{aligned} \tag{204}$$

where Λ_c is the aggregate contention packet arrival rate and t_s is the slot size. To account for a non-zero value for p_s, we must account for the probability that a node will miss a contention slot for which it is the designated destination of a contention packet transmission. The probability of a contention slot is then

$$\begin{aligned} \Pr[\text{cont slot}] = & \Pr[\text{cont pkt arvl in prev slot}] \\ & + \Pr[\text{cont pkt arvl 2 slots prior}] \\ & \quad \times \Pr[\text{no cont pkt arvl in prev slot}] \\ & \quad \times \Pr[\text{a node misses 1 cont slot}] \\ & + \Pr[\text{cont pkt arvl 3 slots prior}] \\ & \quad \times \Pr[\text{no cont pkt arvl in prev 2 slots}] \\ & \quad \times \Pr[\text{a node misses 2 cont slot}] \\ & + \cdots. \end{aligned} \tag{205}$$

The probability of a contention packet arrival in a slot is the same for all slots and is found in Chapter IV to be p_0. The probability of a non-contention packet arrival in a slot is also the same for all slots and is given by $(1-p_0)$. Substituting into (205) and rearranging, we have

$$\begin{aligned}\Pr[\text{cont slot}] &\equiv p_c \\ &= (1-p_0)\left(1+\sum_{i=1}^{\infty}(p_0)^i \Pr[\text{a node misses } i \text{ cont slots}]\right)\end{aligned} \tag{206}$$

and

$$\Pr[\text{non-cont slot}] \equiv p_{nc} = (1-p_c). \tag{207}$$

We will explore the probability of missed contention packets in more detail later. Substituting (207) into (203), we have

$$\begin{aligned}\overline{\Pi} = \frac{t_{mf}}{t_f} + n\Bigg\{ & p_{nc}\frac{t_{wnc}}{t_f}\times\Pr[\text{awake for non-cont slot}] \\ & +(1-p_{nc})\frac{t_{wc}}{t_f}\times\Pr[\text{awake for cont slot}]\Bigg\}.\end{aligned} \tag{208}$$

We now solve for the probabilities that a node will be awake for the non-contention and contention slot as well as the time a node will stay awake in each of these slots.

The probability that a node will be awake for a non-contention slot is the probability that either the node is the slot owner and it has non-contention traffic to transmit or it is the intended receiver for another node that has traffic to transmit in its slot. This is summarized as

$$\begin{aligned}&\Pr[\text{awake for non-cont slot}] = \\ &\quad \Pr[\text{node is slot owner}] \\ &\qquad \times\Pr[\text{node has non-cont packets to xmt}] \\ &\quad +\Pr[\text{node is not slot owner}] \\ &\qquad \times\Pr[\text{slot owner has non-cont packets to xmt}] \\ &\qquad \times\Pr[\text{node is destination}].\end{aligned} \tag{209}$$

If we assume that a node is assigned exactly one slot in a transmission frame, then the probability that a node is a slot owner is uniformly distributed with

$$\Pr\left[\text{node is slot owner}\right] = \frac{1}{n} \text{ and}$$
$$\Pr\left[\text{node is not slot owner}\right] = \frac{n-1}{n}. \tag{210}$$

The probability that a given node has non-contention traffic to transmit is equivalent to the probability that at least one non-contention packet arrived at the node during the prior transmission frame. Assuming Poisson packet arrivals, this is

$$\Pr\left[\text{a node has non-cont packets to xmt}\right] = 1 - e^{-\lambda_{nc} t_f} \tag{211}$$

where λ_{nc} is the per node non-contention packet arrival rate. Substituting (210) and (211) into (209) and simplifying, we have

$$\Pr\left[\text{awake for non-cont slot}\right] = \frac{1}{n}\left(1 - e^{-\lambda_{nc} t_f}\right) \times \left\{1 + \left(n-1\right) \times \Pr\left[\text{node is destination}\right]\right\}. \tag{212}$$

Finally, neglecting the time required to sample for the contention beacon (which is much smaller than either the frame time or the slot time), a node will be awake in a transmission frame long enough to receive the non-contention traffic or, from Figure 77,

$$t_{wnc} = t_s - t_{IFS} - \left(t_{prop} + t_{guard}\right) \tag{213}$$

where t_{prop} is the maximum propagation distance and t_{guard} is the guard band designed to accommodate slot synchronization errors.

The probability that a node will be awake for a contention slot is the probability that it will detect the contention slot as in

$$\begin{aligned} \Pr\left[\text{awake for cont slot}\right] = & \\ & \Pr\left[\text{node has cont pkt to xmt}\right] \\ & + \Pr\left[\text{node has no cont pkt to xmt}\right] \\ & \quad \times \Pr\left[\text{sample for cont beacon}\right] \\ & + \Pr\left[\text{node has no cont pkt to xmt}\right] \\ & \quad \times \Pr\left[\text{not sample for cont beacon}\right] \\ & \quad \times \Pr\left[\text{awake for non-cont slot}\right], \end{aligned} \tag{214}$$

which is, from (212) and the analysis of (206),

$$\begin{aligned}\Pr\left[\text{awake for cont slot}\right] &= p_{xmt} + \left(1-p_{xmt}\right)p_s \\ &+ \frac{\left(1-p_{xmt}\right)\left(1-p_s\right)}{n}\left(1-e^{-\lambda_{nc}t_f}\right) \\ &\times\left\{1+\left(n-1\right)\times\Pr\left[\text{node is destination}\right]\right\}\end{aligned} \tag{215}$$

where

$$\begin{aligned}p_{xmt} &\equiv \Pr\left[\text{node has cont pkt to xmt}\right] \\ &= \left(1-p_0'\right)\left(1+\sum_{i=1}^{\infty}\left(p_0'\right)^i \Pr\left[\text{node misses } i \text{ cont slots}\right]\right)\end{aligned} \tag{216}$$

and

$$p_0' = e^{-\frac{\Lambda_c}{n}t_s}. \tag{217}$$

Note that for $p_s = 1$, $\Pr\left[\text{awake for cont slot}\right] = 1$ as expected. The time a node will stay awake during a contention slot can be broken into two parts: (1) the time a node stays awake during the contention phase of the slot, t_{wc_c}, (which is comprised of the minislots) and (2) the time a node stays awake during the non-contention phase of the contention slot, t_{wc_nc}, (when the slot owner is permitted to transmit its non-contention traffic). From Figure 77, for a contention slot with *k* minislots each of length t_{ms}, the latter is

$$t_{wc\_nc} = \left(t_s - t_b - kt_{ms}\right)\times\Pr\left[\text{awake for non-cont slot}\right]. \tag{218}$$

To calculate the former, we must examine each minislot individually using the slotted ALOHA with periodic server vacation analysis of Section IV.D.

At a minimum, a node will wake up to receive the minislot destination bitmap for every minislot and

$$\min\left(t_{wc\_c}\right) = kt_{mms}. \tag{219}$$

To find the mean value, we must calculate the probability that a node will stay awake for each minislot. This is the probability that either the node will attempt to transmit in this minislot or it will be the destination for another node's successful transmission. In general,

$$\begin{aligned}\Pr[\text{awake for minislot}] = \\ &\Pr[\text{attempt xmsn in minislot}] \\ &+\Pr[\text{not attempt xmsn in minislot}] \\ &\times\Pr[\text{successful xmsn in minislot}] \\ &\times\Pr[\text{node is destination}].\end{aligned} \tag{220}$$

This is a recursive equation, because, from [167], the probability of both an attempted transmission and a successful transmission depend upon the outcome of the prior minislots. We can achieve a closed form solution for the duty cycle if we make the assumption that a node stays awake for all of the minislots (this can clearly be seen to be a conservative estimation for t_{wc_c}). Thus,

$$t_{wc\_c} = kt_{ms}\,. \tag{221}$$

Combining (218) and (221), we have

$$\begin{aligned}t_{wc} = kt_{ms} &+ \left(t_s - t_b - kt_{ms}\right) \\ &\times\Pr[\text{awake for non-cont slot}].\end{aligned} \tag{222}$$

Substituting (212), (213), (215) and (222) into (208), we finally arrive at our result for the mean duty cycle of

$$\begin{aligned}\overline{\Pi} = \frac{t_{mf}}{t_f} + n\Bigg\{ & p_{nc}\,\frac{t_s - t_{IFS} - \left(t_{prop} + t_{guard}\right)}{t_f} \\ &\times\frac{1}{n}\left(1-e^{-\lambda_{nc}t_f}\right)\left(1+\left(n-1\right)\times\Pr[\text{node is dest}]\right) \\ &+\frac{\left(1-p_{nc}\right)}{t_f}\Big[\,p_{xmt} + \left(1-p_{xmt}\right)p_s + \left(1-p_{xmt}\right)\left(1-p_s\right) \\ &\times\frac{1}{n}\left(1-e^{-\lambda_{nc}t_f}\right)\left(1+\left(n-1\right)\times\Pr[\text{node is dest}]\right)\Big] \\ &\times\Big[\,kt_{ms} + \left(t_s - t_b - kt_{ms}\right) \\ &\times\frac{1}{n}\left(1-e^{-\lambda_{nc}t_f}\right)\left(1+\left(n-1\right)\times\Pr[\text{node is dest}]\right)\Big]\Bigg\}.\end{aligned} \tag{223}$$

Checking the limits, the minimum duty cycle represents the case where there is no traffic at all (i.e., $\lambda_{nc} = \Lambda_c = 0$). In this case, $p_{nc} = p_0 = 1$, $e^{-\lambda_{nc}t_f} = 1$ and (223) reduces to

$$\overline{\Pi} = \frac{t_{mf}}{t_f}, \tag{224}$$

(the term that includes kt_{ms} is zeroed out in this case) which is the expected result from (201). For $\lambda_{nc} \to \infty$ with no contention traffic present ($\Lambda_c = 0$), $p_{nc} = p_0 = 1$, $e^{-\lambda_{nc} t_f} = 0$ and (223) reduces to

$$\begin{aligned} \overline{\Pi} = \frac{t_{mf}}{t_f} + \frac{t_s - t_{IFS} - \left(t_{prop} + t_{guard}\right)}{t_f} \\ \times \left(1 + \left(n-1\right) \times \Pr\left[\text{node is dest}\right]\right) \end{aligned} \tag{225}$$

For $\Lambda_c \to \infty$ with no non-contention traffic present ($\lambda_{nc} = 0$), $p_{nc} = p_0 = 0$, $e^{-\lambda_{nc} t_f} = 1$ and (223) reduces to

$$\overline{\Pi} = \frac{t_{mf} + p_s n k t_{ms}}{t_f}. \tag{226}$$

Finally, for the limit in which both $\lambda_{nc} \to \infty$ and $\Lambda_c \to \infty$, $p_{nc} = p_0 = 0$, $e^{-\lambda_{nc} t_f} = 0$ and

$$\begin{aligned} \overline{\Pi} = \frac{t_{mf}}{t_f} + \frac{n}{t_f}\Big[\Big[p_{xmt} + \left(1 - p_{xmt}\right) p_s + \left(1 - p_{xmt}\right)\left(1 - p_s\right) \\ \times \frac{1}{n}\left(1 + \left(n-1\right) \times \Pr\left[\text{node is dest}\right]\right)\Big] \\ \times \Big[kt_{ms} + \left(t_s - t_b - kt_{ms}\right) \\ \times \frac{1}{n}\left(1 + \left(n-1\right) \times \Pr\left[\text{node is dest}\right]\right)\Big]. \end{aligned} \tag{227}$$

If we assume every packet is transmitted to all nodes, $\Pr\left[\text{node is dest}\right] = 1$ and

$$\begin{aligned} \overline{\Pi} = \frac{t_{mf}}{t_f} + n\Bigg\{ p_{nc}\left(1 - e^{-\lambda_{nc} t_f}\right) \frac{t_s - t_{IFS} - \left(t_{prop} + t_{guard}\right)}{t_f} \\ + \frac{\left(1 - p_{nc}\right)}{t_f} \\ \times \Big[p_{xmt} + \left(1 - p_{xmt}\right) p_s + \left(1 - p_{xmt}\right)\left(1 - p_s\right)\left(1 - e^{-\lambda_{nc} t_f}\right)\Big] \\ \times \Big[kt_{ms} + \left(t_s - t_b - kt_{ms}\right)\left(1 - e^{-\lambda_{nc} t_f}\right)\Big]\Bigg\}. \end{aligned} \tag{228}$$

We plot this result with $p_s = 1$ as a function of the non-contention packet arrival rate in Figure 80 and the contention packet arrival rate in Figure 81. As expected, duty cycle increases with increasing packet arrival rate and reflects the bounds derived in (224) through (227). The only exception is in Figure 81, where the duty cycle decreases slightly as λ_{nc} increases for large values of Λ_c. This is because as Λ_c increases, more slots are seized as contention slots and the time a node stays awake during a contention slot is less than that for a non-contention slot. This is due to the added overhead of the contention beacon and is apparent in a comparison of (213) and (222). We also plot the results in Figures 82 and 83 for the case where the mean node degree is three and a packet is randomly transmitted to a single neighbor based on a uniform distribution ($\Pr[\text{node is dest}] = 0.33$). It should be noted that in the cases in which $\Lambda_c = 0$, the protocol does not make use of the contention mode and the plots are not dependent on the assumption used to arrive at (221). For the plots of Figures 80 through 83, the frame size is 1 s, the minislot size is 0.001 s and the beacon time is 0.001 s. There are 5 slots in a frame, 50 minislots in a contention slot and the channel data rate is 1 Mbps. The duty cycle performance is normalized to the frame time, which can therefore be scaled to values more representative of typical wireless sensor applications (usually 1 to 10 ms), as desired.

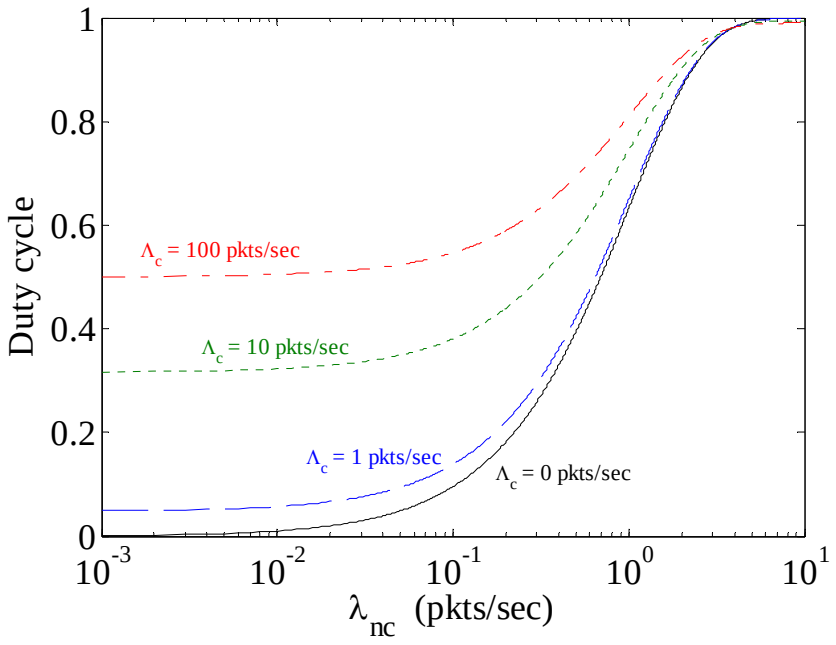


Figure 80. Duty cycle plotted as a function of non-contention packet arrival rates for various values of contention packet arrival rate with $\Pr[\text{node is dest}]=1$.

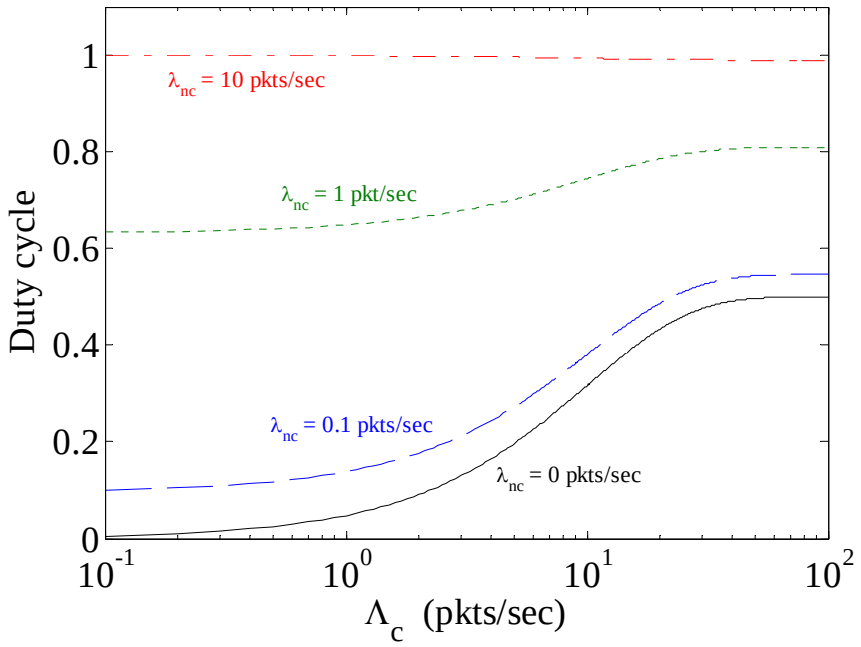


Figure 81. Duty cycle plotted as a function of contention packet arrival rates for various values of non-contention packet arrival rate with $\Pr[\text{node is dest}]=1$.

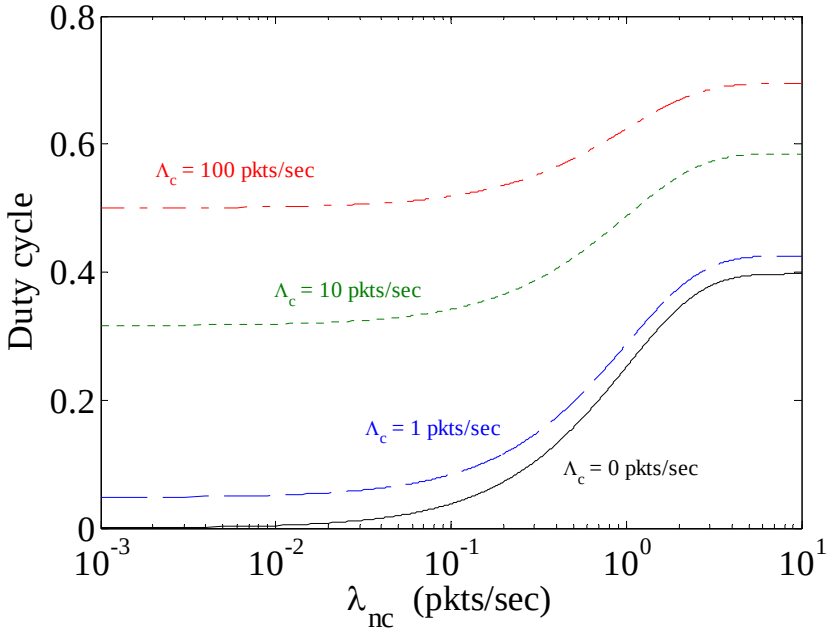


Figure 82. Duty cycle plotted as a function of non-contention packet arrival rates for various values of contention packet arrival rate with $\Pr[\text{node is dest}] = 0.33$.

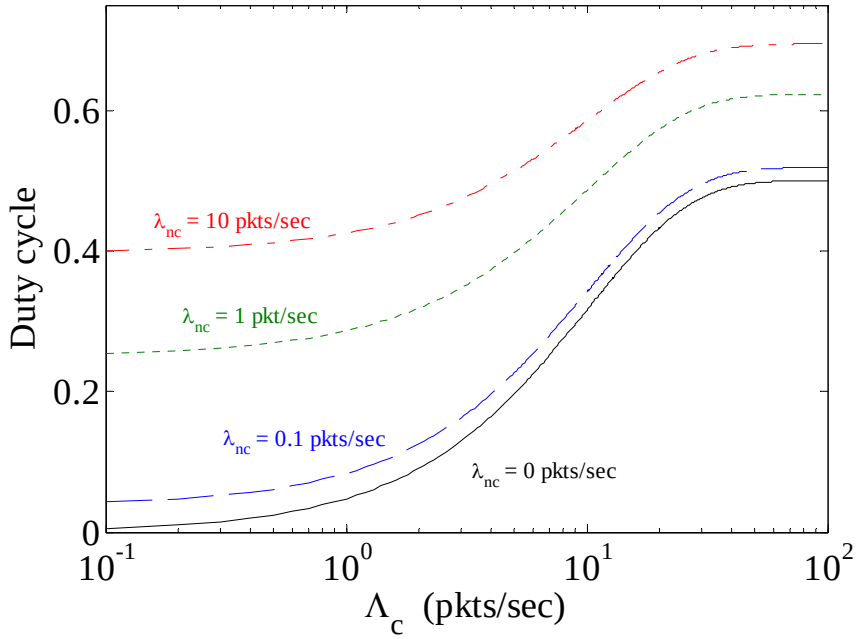


Figure 83. Duty cycle plotted as a function of contention packet arrival rates for various values of non-contention packet arrival rate with $\Pr[\text{node is dest}] = 0.33$.

b. Throughput and Delay

In this section, we derive the throughput and delay for the proposed scheme. We address the non-contention and contention modes individually. These results can be combined to arrival at the overall throughput and delay using the results of Chapter IV.

The throughput and delay for the non-contention mode (S^{nc} and D^{nc}, respectively) of the proposed scheme can be found by accounting for (206) and (207) in the analysis of Chapter IV. The normalized throughput is thus

$$S^{nc} = \begin{cases} \dfrac{\Lambda_{nc}\bar{L}}{R} & \Lambda_{nc} \le S^{nc}_{\max}\left(\dfrac{R}{\bar{L}}\right) \\ S^{nc}_{\max} & \text{otherwise} \end{cases} \tag{229}$$

for

$$S^{nc}_{\max} = \frac{t_{wnc} + (1-p_{nc})(t_{IFS} - t_b - kt_{ms})}{t_s} \tag{230}$$

where $\bar{L}$ is the mean non-contention packet size in bits. The mean delay is

$$\begin{aligned} \bar{D}^{nc} = {} & \frac{t_s}{2} + \frac{\lambda_{nc}\left(\dfrac{1-(1-p_{nc})^{\kappa}}{p_{nc}}\right)^2}{2\left(1-\lambda_{nc}\left(\dfrac{1-(1-p_{nc})^{\kappa}}{p_{nc}}\right)\right)}(1+\Upsilon) \\ & + \left[t_f - t_{wc\_nc}\right]\left(\frac{\kappa(1-p_{nc})^{\kappa+1} - (\kappa+1)(1-p_{nc})^{\kappa} + 1}{p_{nc}}\right) \\ & + \left[\kappa(t_f - t_{wc\_nc}) - t_{IFS} + t_b + kt_{ms}\right](1-p_{nc})^{\kappa} \\ & - t_f + t_{IFS} + t_{nc} + t_{wc\_nc} + t_{prop} \end{aligned} \tag{231}$$

where

$$\Upsilon = \left(\frac{1-p_{nc}}{\left(1-(1-p_{nc})^{\kappa}\right)^2} \right) \times[(1-p_{nc})^{2\kappa} + \left((\kappa-1)^2-1\right)(1-p_{nc})^{\kappa} \cdot - \left(2\kappa^2-2\kappa-1\right)(1-p_{nc})^{\kappa-1} + \kappa^2\left(1-(p_{nc})^2\right)(1-p_{nc})^{\kappa-2}-1], \quad (232)$$

and

$$\kappa = \frac{t_{wnc}}{t_{wc\_nc}}. \quad (233)$$

The throughput and delay for the contention mode (S^c and D^c, respectively) can again be derived from the analysis in Chapter IV. If we make the assumption that all pending contention packets with the exception of those whose destination node is asleep are successfully transmitted in a contention slot, then the repetition of a contention slot only reduces the overall contention mode throughput if it is repeated when no new contention packet has arrived in the previous slot. In this case, rather than calculating the throughput across a single contention slot as in Chapter IV, we must now calculate it across multiple slots as in

$$S^c = \frac{L_c}{R}\sum_{i=0}^{\infty}\left(\frac{Q(0)}{(i+1)t_s}\right)(p_0')^i \Pr[\text{node misses } i \text{ cont slots}] \quad (234)$$

where L_c is the mean contention packet size in bits and $Q(0)$ is the mean number of nodes with a contention packet pending for transmission at the beginning of a contention slot.

Turning to the contention delay, our analysis reflects the observation that the when a node misses a contention slot, it must wait until the next contention slot to receive the intended contention mode packet. Thus for every missed contention slot, the packet delay is increased by t_s and the mean is thus

$$D^c = D^c_{p_s=1} + \sum_{i=1}^{\infty} i t_s \Pr[\text{node misses } i \text{ cont slots}] \quad (235)$$

where $D^{c}_{p_s=1}$ is the delay associated with the contention mode for $p_s = 1$ (i.e., $\Pr[\text{missed cont slot}] = 0$), which can be found numerically as in Chapter IV.

c. Effect of Preamble Sampling Parameter

We begin by discussing the effect of the preamble sampling parameter p_s on throughput and delay and then look at the duty cycle. We conclude by highlighting the role it plays in trading off performance and the energy efficiency. To clearly see the role of p_s, we return to the probability of a missed contention slot. A node will sleep through a contention slot if (1) it is not scheduled to be awake during the original non-contention slot that was redesignated as a contention slot, (2) it does not have any contention traffic to transmit and (3) it does not sample for a contention beacon. Thus, from (215),

$$\begin{aligned}
&\Pr[\text{a node misses a cont slot}] \\
&\quad = \left(1 - \Pr[\text{awake for cont slot}]\right) \\
&\quad = 1 - \Big\{ p_{xmt} + \left(1 - p_{xmt}\right) p_s \\
&\qquad + \frac{\left(1 - p_{xmt}\right)\left(1 - p_s\right)}{n}\left(1 - e^{-\lambda_{nc} t_f}\right) \\
&\qquad \times \left(1 + \left(n-1\right) \times \Pr[\text{node is destination}]\right)\Big\}.
\end{aligned} \tag{236}$$

Since $\lambda_{nc} t_f$ is positive, n is a positive integer and $\Pr[\text{node is destination}]$ is less than one, the term $\frac{1}{n}\left(1 - e^{-\lambda_{nc} t_f}\right)\left(1 + \left(n-1\right) \times \Pr[\text{node is destination}]\right)$ is less than one for all values of n, λ_{nc}, t_f, and $\Pr[\text{node is destination}]$ and, therefore,

$$\begin{aligned}
1 - p_{xmt} > &\frac{1 - p_{xmt}}{n}\left(1 - e^{-\lambda_{nc} t_f}\right) \\
&\times \left(1 + \left(n-1\right) \times \Pr[\text{node is destination}]\right).
\end{aligned} \tag{237}$$

Thus, as p_s increases, the $\Pr[\text{a node misses a cont slot}]$ increases and, from (206) and (207), p_{nc} decreases. From the results in the previous section, we can see that non-contention mode throughput decreases and non-contention delay increases with decreasing p_{nc}. We also see that the non-contention throughput also decreases and

contention delay also increases with increasing $\Pr[\text{a node misses a cont slot}]$. Therefore, throughput and delay performance in both modes drops off as p_s increases.

To see the effect of p_s on duty cycle (and, therefore, energy consumption), we plot (223) as a function of p_s in Figures 84-86. As expected, in most cases, we see that the duty cycle strictly decreases (and hence the energy consumption decreases) with decreasing p_s. There are two exceptions to this observation in which the duty cycle appears to remain constant with respect to p_s. The first can be seen in Figures 84 and 85 when the contention packet arrival rate is zero. In this case, the duty cycle is clearly independent of p_s. This is also reflected in (223), which reduces to

$$\begin{aligned}\overline{\Pi} = \frac{t_{mf}}{t_f} + n\Bigg\{ & \frac{t_s - t_{IFS} - \left(t_{prop} + t_{guard}\right)}{t_f} \\ & \times \frac{1}{n}\left(1 - e^{-\lambda_{nc} t_f}\right)\left(1 + \left(n-1\right) \times \Pr[\text{node is dest}]\right)\Bigg\}\end{aligned} \tag{238}$$

because $e^{-\Lambda_c t_f} = 1$, $p_0 = 1$, $p_c = 0$, and, thus, $p_{nc} = 1$ for $\Lambda_c = 0$ from (204), (206), and (207). The second exception can be seen in Figure 86 where the non-contention packet arrival rate is very large and the non-contention mode is saturated. Despite the minimal impact of a change in p_s in this case, the duty cycle is a function of p_s (with the above-mentioned exception for the case in which $\Lambda_c = 0$) since (223) reduces to

$$\begin{aligned}\overline{\Pi} = \frac{t_{mf}}{t_f} + n\Bigg\{ & p_{nc}\frac{t_s - t_{IFS} - \left(t_{prop} + t_{guard}\right)}{t_f} \times \frac{1}{n}\left(1 + \left(n-1\right) \times \Pr[\text{node is dest}]\right) \\ & + \frac{\left(1 - p_{nc}\right)}{t_f}\Big[p_{xmt} + \left(1 - p_{xmt}\right) p_s + \left(1 - p_{xmt}\right)\left(1 - p_s\right) \\ & \times \frac{1}{n}\left(1 + \left(n-1\right) \times \Pr[\text{node is dest}]\right)\Big] \\ & \times \Big[k t_{ms} + \left(t_s - t_b - k t_{ms}\right) \\ & \times \frac{1}{n}\left(1 + \left(n-1\right) \times \Pr[\text{node is dest}]\right)\Big]\Bigg\}.\end{aligned} \tag{239}$$

because $e^{-\lambda_{nc} t_f} = 0$ as $\lambda_{nc} \to \infty$.

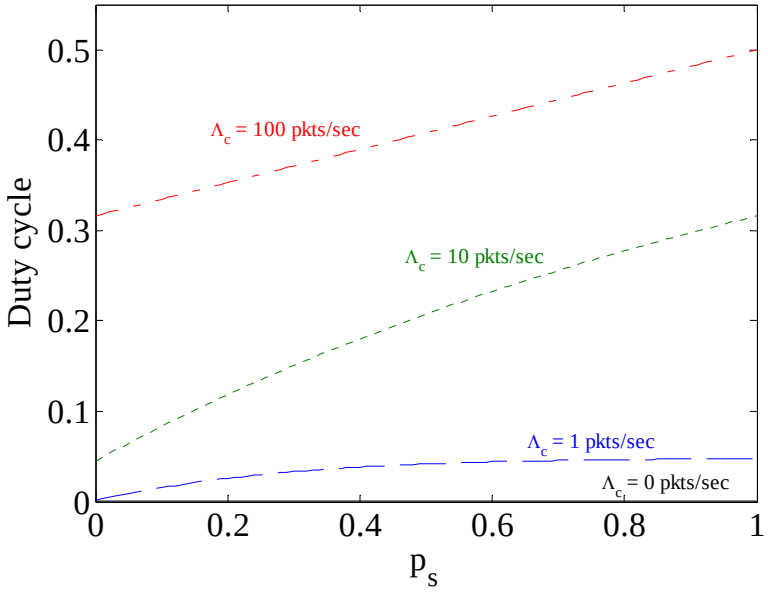


Figure 84. Duty cycle plotted as a function of the sampling probability for various values of contention packet arrival rate with a non-contention packet arrival rate of 0.0 pkts/sec and $\Pr[\text{node is dest}]=1$.

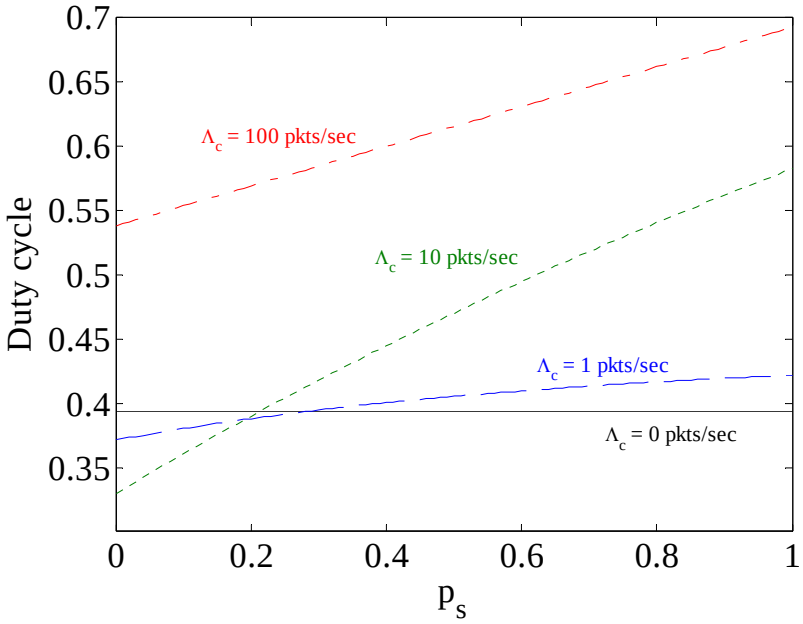


Figure 85. Duty cycle plotted as a function of the sampling probability for various values of contention packet arrival rate with a non-contention packet arrival rate of 0.5 pkts/sec and $\Pr[\text{node is dest}]=1$.

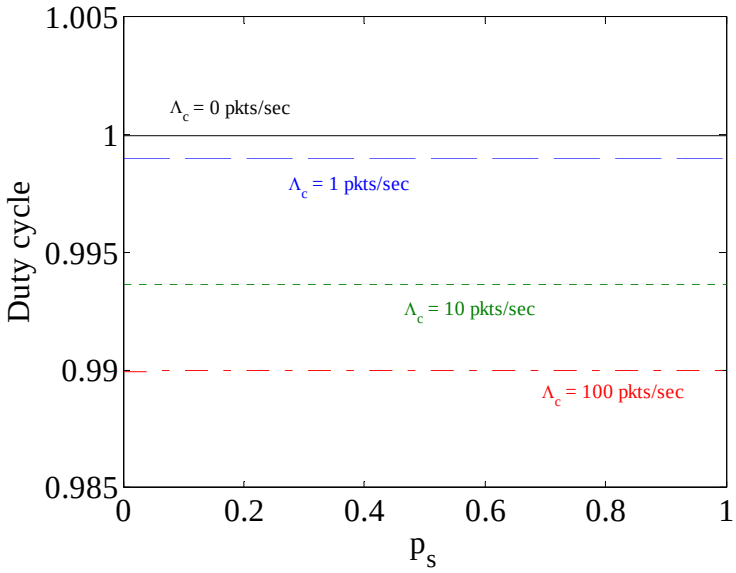


Figure 86. Duty cycle plotted as a function of the sampling probability for various values of contention packet arrival rate with a non-contention packet arrival rate of 10 pkts/sec and $\Pr[\text{node is dest}]=1$.

In general, then, it can be seen that when contention flow traffic is non-zero and the non-contention mode is not saturated, the parameter provides p_s provides a mechanism to trade-off delay and throughput performance for energy efficiency. As p_s is decreased, throughput decreases and delay increases, but energy consumption (as captured by the duty cycle) decreases. Alternately, as p_s is increased, throughput increases and delay decreases, but energy consumption increases.

3. Simulation Results

We now compare the analysis of the previous section with simulation results obtained using the OPNET® Modeler Suite. The simulation includes 10 nodes with a channel data rate of 1 Mbps, a frame length of 1 s, a beacon duration and minislot size of 1 ms, and a data interframe space of 0.1 ms. From these inputs, the frame destination bitmap size is 0.1 ms and the slot size is 99.99 ms. The per node non-contention packet arrival rate is allowed to vary from zero to 10 packets/sec for various values of the aggregate contention packet arrival rate.

The frame time value has been selected to reduce simulation run time but can be scaled to values more representative of typical wireless sensor applications (1 to 10 ms) because the duty cycle performance is normalized to this frame time.

We examine both the steady state and the transient results. The former are compared with the analysis of the previous section for $p_s = 1$ and $\Pr[\text{node is dest}] = 1$ (see Figure 82) in Figure 87. The duty cycle seen in the simulations closely follows that from (223) for all values of λ_{nc} and Λ_c.

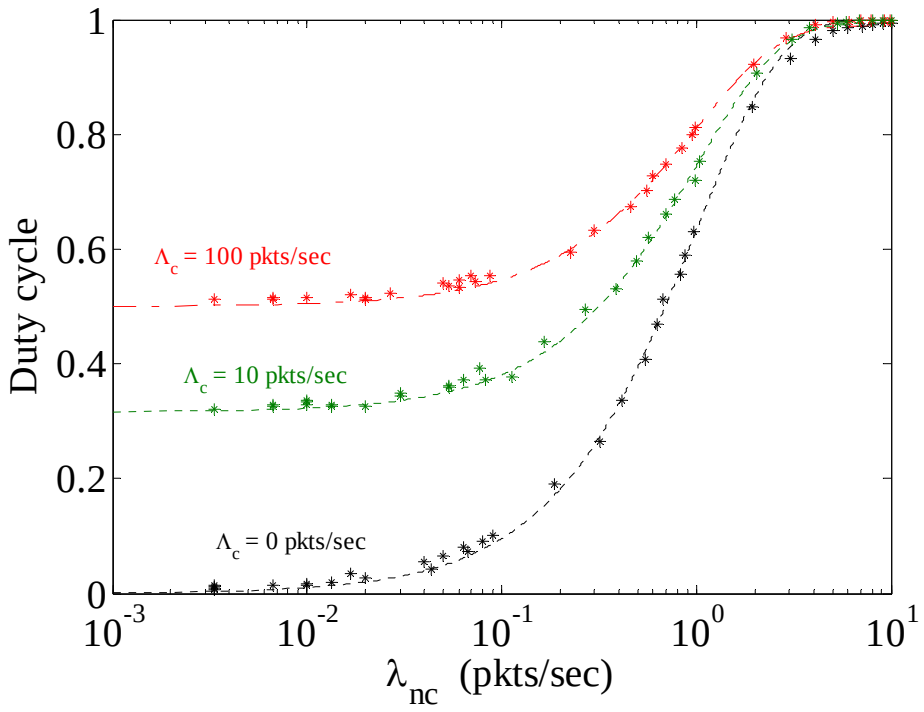


Figure 87. Comparison of the analysis in the previous section (dashed lines on plot) with OPNET® simulation results (discrete points on plot) for the duty cycle plotted as a function of non-contention packet arrival rates for various values of contention packet arrival rate with $\Pr[\text{node is dest}] = 1$.

The transient results are provided in Figure 88. In this plot, the control flow remains constant at 0.1 pkts/sec while the data packet flow is increased from zero to more than 50 pkts/sec. As a function of time, we plot the duty cycle on the left axis and the per node data packet arrival rate on the right axis. We also include the points at which individual nodes transition the data packet flow from the contention to the non-contention

mode. We can see three regions of operation emerge as the data packet arrival rate is increased. In the first region, all of the nodes are transmitting the data packets in the contention mode. Here, the duty cycle is defined by the contention mode traffic flow and its performance mirrors that in the $\lambda_{nc} = 0$ curve in Figure 81. In this region, the duty cycle reaches a maximum defined by (226). In the next region, nodes begin to transition the data packet flow to the non-contention mode and the duty cycle rises in increments as nodes complete the transition. Finally, all nodes will have completed the transition and the duty cycle rises towards the maximum value in Figure 80.

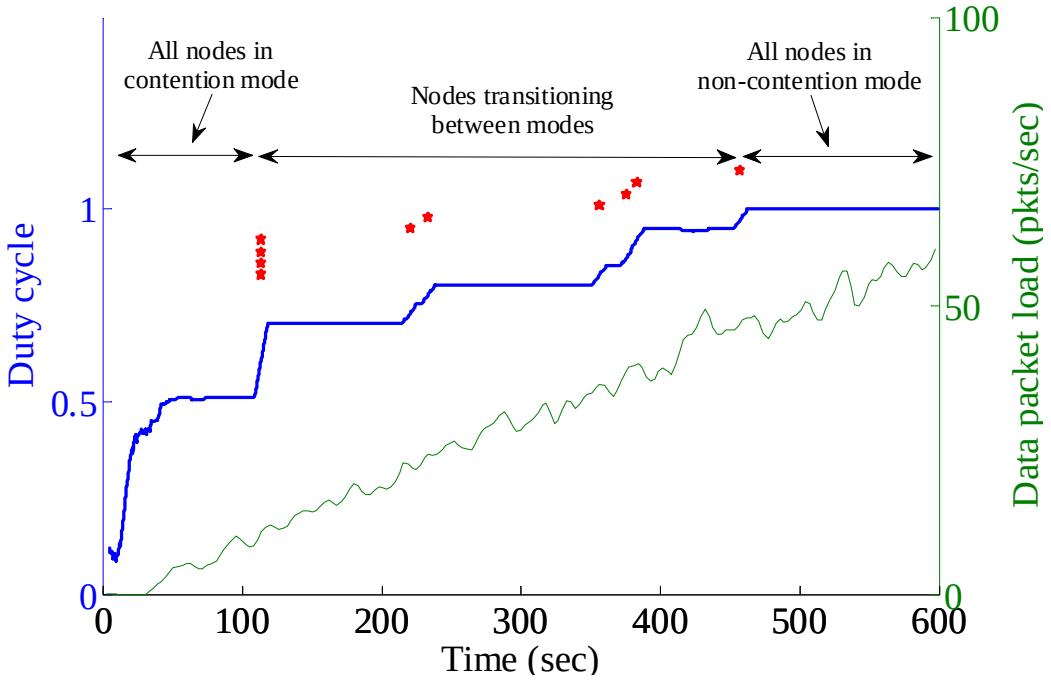


Figure 88. Transient results for duty cycle as a function of increasing data packet arrival rate. The red stars indicate points at which a node transitioned the data packet flow from the contention to the non-contention mode. The control packet arrival rate is constant at 0.1 pkts/sec.

In this chapter, we began by analyzing the energy consumption of centralized and distributed solutions in multi-hop wireless sensor networks. We quantified the energy-efficiency of the preamble sampling technique in the context of these solutions and applied it to the results of Chapters III and IV to propose an energy-efficient, flow-specific medium access scheme based on an adaptive sleep and wake cycle using

preamble sampling. We provided analysis to show that the duty cycle (and hence the energy consumption) of the proposed scheme decreases as the packet arrival rate decreases. We also introduced a preamble sampling probability parameter that was shown to be capable of managing the trade-off between delay and throughput and energy efficiency. Simulation results were provided to validate the analysis and together they demonstrate that our proposed scheme approaches the low duty cycles needed to support the rigorous demands of energy-constrained wireless sensor network applications. Having now developed an energy-efficient implementation for small-scale wireless sensor networks, we now turn our attention to the novel application of large-scale wireless sensor networks in the next chapter.

VI. FLOW-SPECIFIC MEDIUM ACCESS FOR NETWORKED SATELLITE SYSTEMS

Having effectively (in Chapter IV) and efficiently (in Chapter V) implemented the groundbreaking traffic-adaptive, flow-specific medium access approach introduced in Chapter III, we now turn our attention to a novel application of traffic-adaptive CWS-MAC. This application is inspired by the system-of-systems approach that views a ballistic missile defense system (BMDS) as a large-scale, wireless sensor network [169],[170] (such as that shown in Figure 89). This viewpoint implicitly relies on an effective and efficient underlying medium access scheme that can support the large propagation delays encountered in these networked satellite systems. By the term *networked satellite system*, we refer to a satellite-based communication system in which terrestrial, terrestrial-satellite and inter-satellite links co-exist and the satellites are capable of serving as the equivalent of network routers rather than simply acting as communication relays between two (or more) earth stations. Existing networked satellite systems that include inter-satellite links (e.g., Iridium [171]) typically utilize FDMA/TDMA terrestrial-satellite links and dedicated inter-satellite links. However, a number of proposals have been made to adopt commercial-off-the-shelf medium access solutions for inter-satellite communication within LEO satellite formations including recent efforts focused on the IEEE 802.11 contention-based standard [172]-[176]. In the majority of these proposals, though, one medium access scheme is dedicated to the inter-satellite links, while another, different, scheme is proposed for the terrestrial-satellite links.

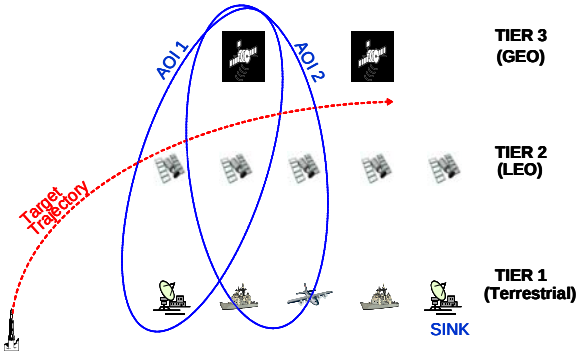


Figure 89. Large-Scale Wireless Sensor Network (From [169]).

The major contribution of this chapter is the proposal of a single protocol, flow-specific medium access solution for networked satellite systems that is capable of dynamically responding to changes in both flow and link characteristics. In LEO satellite networks, the propagation distances (and, hence, the propagation delays) along the inter-satellite links are often equivalent to or even greater than those found on the terrestrial-satellite links [171]. This leads one to consider a single, integrated medium access solution that optimizes network performance (specifically, throughput and delay) for the traffic found on both sets of links. We quantify the effect of large propagation distances on both contention-based and contention-free solutions and achieve improved delay performance by proposing a medium access scheme that dynamically adapts to changing flow and link conditions.

This chapter is organized as follows. We begin by reviewing related existing work in the first section. We then demonstrate the effect of propagation distance on medium access solutions for networked satellite systems and explore the nature of the traffic flows residing in these networks including their performance in light of existing medium access solutions. A traffic-adaptive, flow-specific medium access solution for networked satellite systems is proposed and we conclude by providing simulation results.

A. NETWORKED SATELLITE SYSTEMS

One of the earliest examples of a networked satellite system, the Iridium satellite system was deployed in 1998 and has been widely studied [171],[177]. This global voice and data communications system is comprised of 66 LEO satellites in six distinct polar planes that each contain 11 satellites at an altitude of 780 km (an in-orbit spare was used to replace a satellite that was lost due to a well-publicized collision with an out-of-service Russian satellite in February of 2009). Permanent inter-satellite links exist between neighboring satellites within the same plane (referred to as *intraplane* links) while dynamic links exist between satellites on different planes (*interplane* links) when the orbits permit. The intraplane propagation distances are fixed at 4030 km and the interplane propagation distances vary between 3270 km and 4480 km [177]. The Iridium uplinks and downlinks are supported by a combination of FDMA and TDMA in which

240 41.67 kHz channels occupy a 10.5 MHz bandwidth in the L-band (specifically, 1616 MHz to 1626.5 MHz) with approximately 2 kHz of guard band between channels [171]. Three phased-array antennas support 48 spot beams per satellite. The TDMA scheme is implemented using a 90 ms frame that includes four uplink and four downlink slots of 8.64 ms each. Framing and guard slots are provided in the remaining 20.88 ms. The burst data rate is 50 kb/s, which can support four full-duplex channels at 4800 b/s. Each Iridium satellite can also support up to four inter-satellite links through the use of four dedicated onboard antennas. These links operate at 25 Mb/s utilizing a connection-based switching protocol similar to Asynchronous Transfer Mode (ATM) [176]. Propagation delays for the uplinks and downlinks of the Iridium system are 2.05 ms and inter-satellite propagation delays are approximately 13.33 ms [171]. Simulation results have been used to explore end-to-end packet delay for LEO satellite constellations with and without inter-satellite links [178] and end-to-end propagation delays in the Iridium system have been estimated to be on the order of 100 to 210 ms [171].

Recently, a number of researchers have explored the use of IEEE 802.11 in networked satellite systems. The performance of IEEE 802.11 on inter-satellite links has been compared to both ATM [172] and a wireless version of the IEEE 1394 serial interface [173]. Simulation results provided indicate that while overall throughput is comparable for both IEEE 802.11 and ATM, queuing, processing and end-to-delay packet delay is lower for the IEEE 802.11 configuration [172]. In contrast, additional simulation results seem to indicate that wireless IEEE 1394 outperforms IEEE 802.11 in terms of queuing delay [173], but the potentially long propagation distances associated with actual satellite formations do not appear to be modeled.

Given these results, commercial-off-the-shelf IEEE 802.11 [174] and a modified version of IEEE 802.11 [175],[176] have been proposed for the inter-satellite links of upcoming picosatellite missions. Picosatellites are small (less than 1 kg) satellites that are often designed around the CubeSat platform [179]. Modifications to the existing off-the-shelf IEEE 802.11 implementation [175],[176] include the redefinition of both the timing parameters and the minimum contention window size to accommodate the large propagation distances. The slot time, DCF Inter-frame Space (DIFS), and AckTimeout

are all adjusted to account for the increased propagation delay and nodes are allowed to dynamically adjust their minimum contention window size based on the observed ratio of unsuccessful packet transmissions to successful packet transmissions.

B. EFFECT OF PROPAGATION DISTANCE ON MEDIUM ACCESS IN NETWORKED SATELLITE SYSTEMS

The effect of offered load on both throughput and delay performance of contention-based medium access including CSMA and its derivative IEEE 802.11 has been well-studied [21],[180]-[182]. Throughput and delay performance of contention-based schemes is also dependent on propagation distance [21]. While in most existing work, the emphasis is on offered load vice propagation distance, it has been shown that performance of these approaches also drops off as propagation distance increases. In contrast, the contention-free approaches are less sensitive to large propagation distances and, accordingly, have often been the approach of choice for satellite-based systems [171],[177].

The mean access delay for slotted non-persistent CSMA with binary exponential backoff (upon which IEEE 802.11 is based) is [180]

$$\bar{D}_{CSMA} = \frac{t_{pkt}}{2}\left[\frac{aWp_s}{1-2(1-p_{succ})} + \frac{2+5a}{p_{succ}} - \frac{(2+4a)\dfrac{1-e^{-aG}}{1+a-e^{-aG}}}{p_s} - a(4+W)\right], \qquad p_{succ} > 0.5 \tag{240}$$

where W is the initial window size, G is the offered load (consisting of both transmitted and retransmitted packets), p_{succ} is given by

$$p_{succ} = \frac{ae^{-aG}}{1+a-e^{-aG}} \tag{241}$$

and a is the maximum propagation delay normalized by the packet transmission time or

$$a = \frac{t_{prop}}{t_{pkt}}. \tag{242}$$

In contrast to (240), the access delay of TDMA is independent of propagation distance as in [26]

$$\bar{D}_{TDMA} = \frac{nt_s}{2} + t_s + \frac{\lambda (nt_s)^2}{2(1-\lambda nt_s)} \tag{243}$$

where n is the number of slots in a frame, λ is the packet arrival rate and t_s is the slot size.

The dependence of the delay performance of contention-based schemes such as slotted non-persistent CSMA on both load and propagation distance can be clearly seen in the plot of Figure 90. However, these contention-based approaches are seen to outperform TDMA at low network loads despite the large propagation distances. For this plot, packet transmission time is 1 ms, the CSMA plots assume steady state and the TDMA plot assumes 1 ms slots with 100 slots per frame.

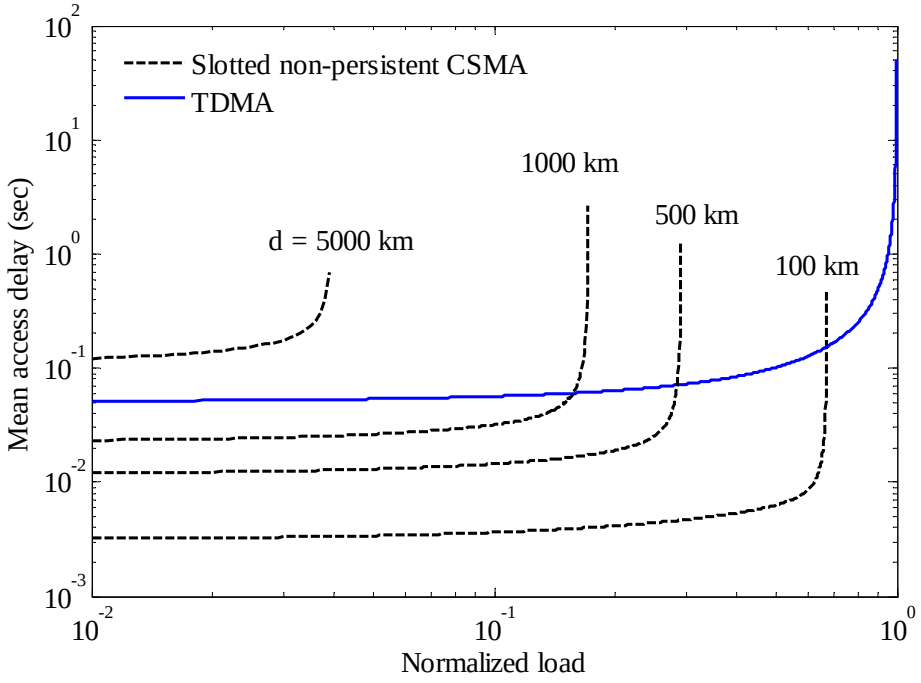


Figure 90. Delay plotted as a function of normalized load for various maximum propagation distances.

C. FLOW-SPECIFIC MEDIUM ACCESS FOR NETWORKED SATELLITE SYSTEMS

In the existing work discussed in Section A, different medium access mechanisms are used for the inter-satellite links and the terrestrial-satellite links. This practice typically results in increased hardware with a larger footprint and increased weight requirements. This is particularly important given the size and weight constraints associated with picosatellites. It is envisioned that these satellites will be flown in master-slave configurations such as the triangular and circular formations in [176]. Here, the inter-satellite distances can range from hundreds to several thousands of kilometers. In light of the analysis of Section B, rather than fixing the medium access scheme on a per link basis, we propose to dynamically adapt it to both the variable load and the changing propagation distances.

We propose a single medium access solution that provides a common access channel for terrestrial, terrestrial-satellite, and inter-satellite links. To achieve this, the proposed mechanism must not only adapt to changes in load and inter-satellite distances, it must also support the multiple flows that exist within the satellite system. For the purposes of this discussion, we identify two representative flows: a data packet flow and a control packet flow. A networked satellite system such as BMDS will be comprised of multiple sensors of varying types that can generate large data rates [169],[170]. For example, while the data rate associated with a radio frequency sensor may be on the order of 192 kbps, the data rate of an infrared sensor can be on the order of 65 Mbps [170]. These high demand data flows compete directly with loss-intolerant spacecraft and sensor control information that is often subject to strict delay bounds.

Traffic-adaptive, flow-specific medium access is well suited to meet the needs of these traffic flows. The traffic-adaptive mechanism is queue based which responds to changes in packet delay, allowing for adaptation to per-link variations in both load and propagation distance. The low demand, delay-sensitive traffic can be dynamically assigned to a contention-based mode while the high demand, throughput-limited data packet flow can be assigned to a non-contention mode. The control traffic will be

"protected" from the data traffic and will continue to meet the strict delay bounds despite potential saturation of the network. By protected, we mean that contention-based flow performance is insulated from the non-contention flow. If properly configured, the maximum achievable data throughput will be higher than the contention-based proposals of [172]-[176] while the delay performance of the control packet flow will be better than a TDMA solution such as that included in the Iridium system [171].

Furthermore, given a fixed data rate, the maximum achievable throughput in a TDMA scheme is limited by the overhead associated with the framing and guard periods. As the size of the transmission slots (and, hence, the frame size) increases, the percentage of overhead decreases and the maximum achievable throughput increases. The size of the frame is bounded by the desired maximum latency because a node must wait for its slot within a frame to transmit. By employing a flow-specific medium access scheme, delay-sensitive flows can be assigned to a contention-based mode, which allows the overall frame size (and, hence, the individual slot size) to be increased to reduce the percentage of per frame overhead and support higher throughput performance in the non-contention mode.

D. PROPOSED MEDIUM ACCESS SCHEME FOR NETWORKED SATELLITE SYSTEMS

We propose to use the traffic-adaptive CWS-MAC protocol to meet the requirements of networked satellite systems medium access. As noted in Chapter IV, user selectable parameters for traffic-adaptive CWS-MAC include the slot size, t_s, the minislot size, t_{ms}, the number of minislots, k, and the lengths of the control beacon and interframe space. To optimize throughput and delay performance in a networked satellite system, these parameters must be tuned to accommodate the large propagation distances associated with a LEO satellite constellation. The interframe space, t_{IFS}, must be greater than the maximum propagation time in the satellite network to allow the control beacon to be detected by all potential slot owners during a control slot. This maximum propagation time will be driven by the inter-satellite links and, as mentioned earlier, is on the order of tens of milliseconds. The interframe space must also be smaller than the sum

of the beacon duration, t_b, and the minimum propagation delay in the network to ensure that the beacon signal is detected prior to its termination. The guard band at the end of a slot must also be increased to account for the larger propagation delays. Finally, the minislot size must be increased to account for the maximum propagation delays of the control packets and the accompanying acknowledgement packets. This will result in either fewer minislots per slot or, alternatively, a larger slot size.

E. SIMULATION RESULTS

Simulation results were generated using OPNET® Modeler for a network comprised of three LEO satellites and two ground stations. Motivated by the Iridium system, the satellites were modeled at an altitude of 780 km with inter-satellite links of up to 4000 km. The simulation parameters used for the traffic adaptive CWS-MAC protocol are shown in Table 8.

Interframe space, t_{IFS}	0.05 sec
Maximum propagation delay, t_{prop}	0.02 sec
Guard time, t_{guard}	0.001 sec
Beacon period, t_b	0.1 sec
Slot length t_s	1 sec
Minislot length, t_{ms}	0.05 sec
Minislots per slot, k	10
Frame length, t_f	5 sec
Slotted ALOHA transmit probability, p	0.3
Channel data rate	1 Mbps
Transition threshold	100 packets

Table 8. Traffic adaptive CWS-MAC parameters used in Section E.

Following the discussion of Section C, two distinct flows were modeled. As shown in Figure 91, the first flow represents a control packet flow that is maintained at a

constant arrival rate of 1 packet/sec (per node) while the second represents the data packet flow and is allowed to vary from 1 packet/sec up through network saturation. Packet size was fixed at 100 bits for the control flow and 900 bits for the data flow. Packet arrivals were Poisson distributed.

Mean delay is plotted in Figure 92 as a function of aggregate load. At low contention levels, both flows remain in the contention-mode of traffic-adaptive CWS-MAC and, therefore, benefit from the better delay performance. As the mean queue size in the contention mode (shown in Figure 93) begins to rise with the increasing data flow arrival rate and the rising packet loss ratio, the mean delay also increases. Upon reaching the queue-based threshold (100 packets in this case), the data flow is transitioned to the non-contention mode and mean delay is recovered for both flows. As seen in Figure 92, the mean delay of the control flow continues to reflect the better performance in the contention-mode, while the data flow now reflects the increased delay associated with the non-contention mode. At very high arrival rates, the data flow saturates the network and its delay performance rapidly deteriorates. The control flow, however, remains protected in the contention-mode and its delay performance remains stable despite the heavy load on the network. The delay performance advantage gained by leaving the control flow in the contention mode is a function of the number of nodes in the network.

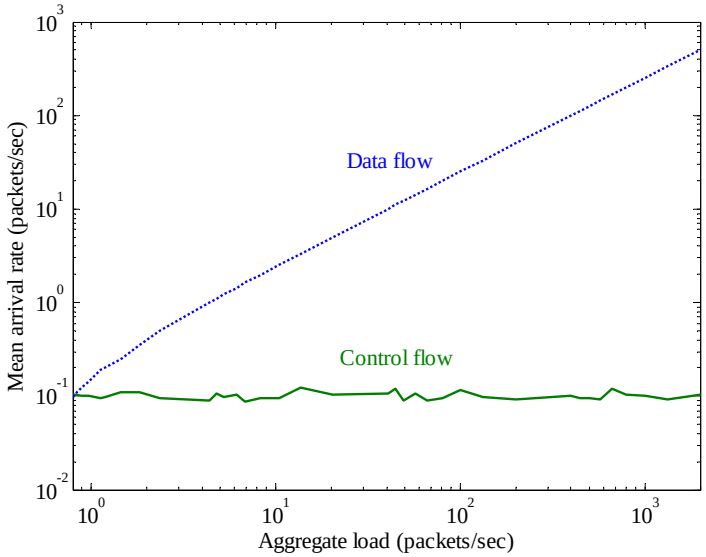


Figure 91. Mean packet arrival rate per node for both the control and data flows plotted as a function of the aggregate load.

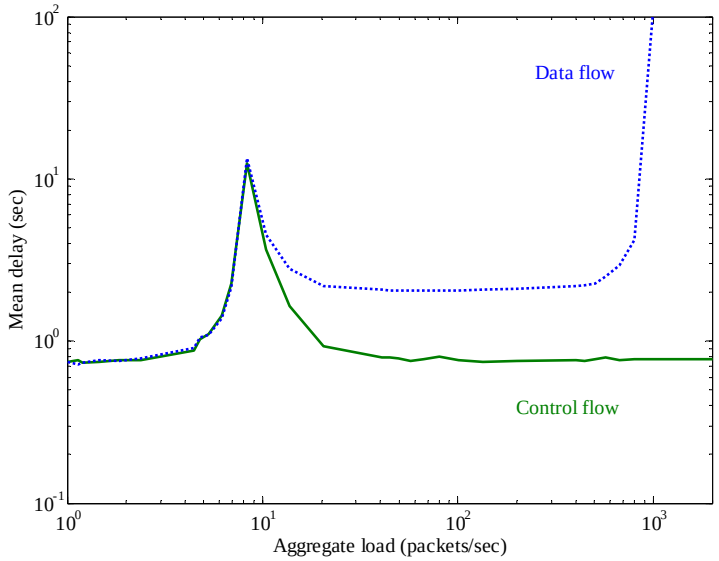


Figure 92. End-to-end delay for both the control and data flows plotted as a function of the aggregate load for a network of two ground stations and three LEO satellites.

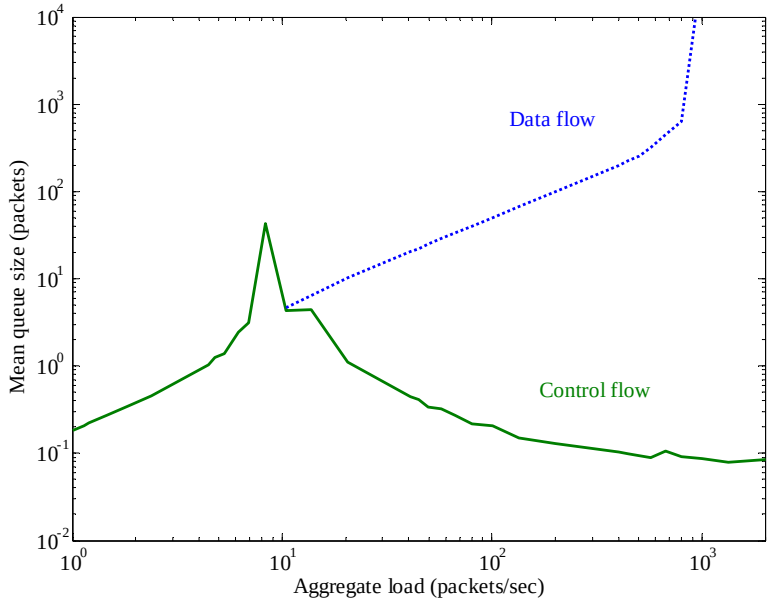


Figure 93. Mean queue size for both the control and data flows plotted as a function of the aggregate load for a network of two ground stations and three LEO satellites.

End-to-end delay is plotted for both flows and compared to that for slotted, non-persistent CSMA and TDMA in Figure 94. As expected, the flow-specific medium access approach outperforms TDMA at low aggregate loads, but does not perform as well as

CSMA due to the underlying slot structure in the flow-specific scheme and the slotted ALOHA implementation of the contention-based mode for traffic-adaptive CWS-MAC. The performance in the contention mode could potentially be improved by utilizing a CSMA-based implementation of CWS-MAC. As the aggregate load is increased, the end-to-end delay performance of the flow-specific approach is seen to be as good as TDMA and significantly better than CSMA. The earlier fall-off of the flow-specific approach when compared to TDMA at high aggregate loads is due to the overhead of the inter-frame space and the presence of the protected contention-based flow.

Mean throughput is plotted in Figure 95 as a function of aggregate load. Data-flow throughput rises until it reaches network saturation at which point it levels off at the maximum throughput associated with the non-contention mode. Control flow throughput is maintained (within the contention mode) despite the fact that the data flow has saturated the non-contention mode.

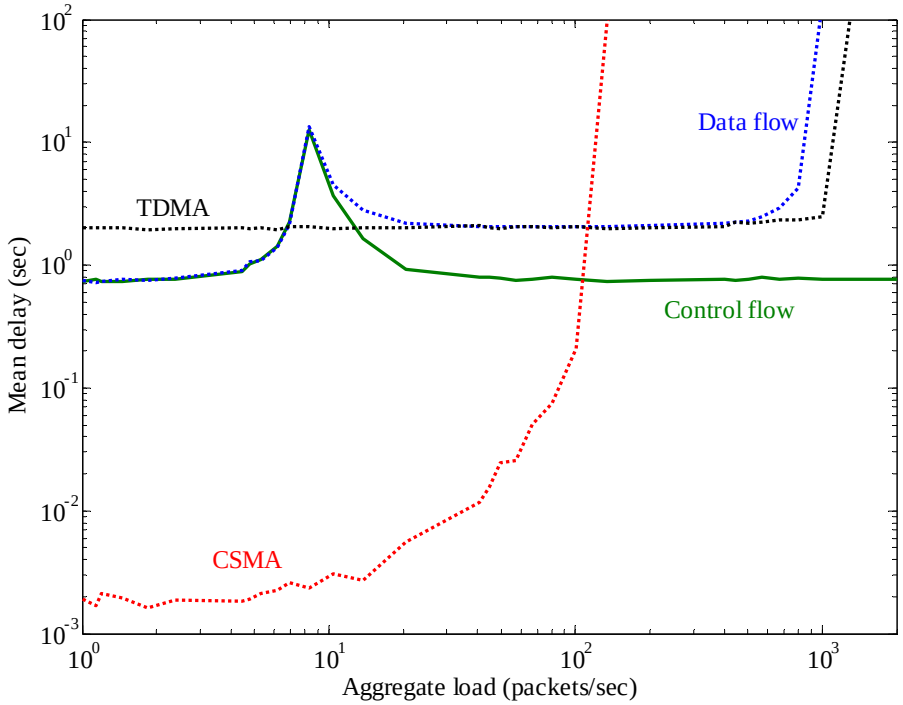


Figure 94. End-to-end delay comparison with CSMA and TDMA for control and data flows for a network of two ground stations and three LEO satellites.

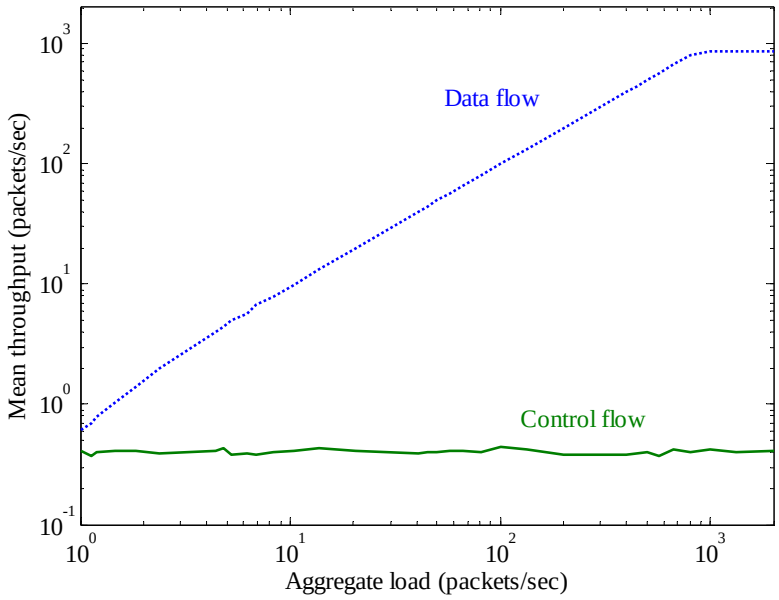


Figure 95. Throughput for both the control and data flows plotted as a function of the aggregate load for a network of two ground stations and three LEO satellites.

In this chapter, we achieved improved delay performance in networked satellite systems by proposing a single protocol, flow-specific medium access solution that is capable of dynamically adapting to changes in both individual flow and link characteristics. The effect of the large propagation distances associated with these satellite-based systems on the performance of both contention-based and contention-free medium access solutions was quantified and the traffic-adaptive CWS-MAC protocol was modified to accommodate the associated large propagation delays. Simulation results were included to demonstrate the improved performance of this flow-specific approach relative to both CSMA- and TDMA-based solutions.

VII. CONCLUSIONS

Wrapping up, this final chapter provides a summary of the research accomplished, discusses the significant contributions of the work, and outlines some potential follow-on research problems.

A. SUMMARY OF RESEARCH

This research began with an investigation into medium access solutions for a distributed radar system [184]. Examining the traffic within the application, we found it to be spatially and temporally dynamic and composed of multiple, distinct, and identifiable flows. Recognizing that this observation and the associated medium access requirements could be extended to wireless sensor networks in general, we postulated that throughput and delay performance could be improved over existing medium access solutions by providing medium access service on a per flow basis.

To begin solving this challenging research problem, we introduced the novel concepts of flow-specific medium access and traffic-adaptive, flow-specific medium access [18],[168]. Based on these ideas, we proposed the Cooperative Wireless Sensor Medium Access Control (CWS-MAC) protocol, a flow-specific medium access scheme that was capable of providing contention or non-contention medium access to multiple flows simultaneously on a per flow basis [18]. To address the dynamic nature of the traffic, we proposed a queue-based traffic observation mechanism and applied it to CWS-MAC [168]. We conducted thorough performance analysis of this proposed scheme [185] that included the development of a general model for traffic-adaptive, flow-specific medium access [168] and the first delay and throughput analysis for a slotted ALOHA system with periodic server vacations [167].

We next turned our attention towards the challenging power-constraints associated with a wireless sensor network implementation of our proposed traffic-adaptive, flow-specific medium access scheme. We began by comparing energy-efficiency of a centralized versus distributed solutions within wireless sensors networks

and found that preamble sampling could be utilized to greatly reduce the energy consumption of the typically power-intensive centralized approaches [186]. Building upon this, we modified our initial proposed scheme to now include preamble sampling and demonstrated that this allowed our traffic-adaptive, flow-specific scheme to achieve the low duty requirements of current wireless sensor network applications [187].

As a capstone to this research effort, we proposed the use of our flow-specific scheme as the basis for the medium access solution for networked satellite systems including the Ballistic Missile Defense System [188]. We found that it is capable of effectively (and efficiently) addressing traffic requirements by dynamically adapting to changes in both individual flow and link characteristics [189].

B. SIGNIFICANT CONTRIBUTIONS

This section discusses the significant contributions of this research. It begins with the initial findings regarding the nature of the traffic in a wireless sensor network and the introduction of the novel concepts of flow-specific medium access and traffic-adaptive, flow-specific medium access. It then highlights the proposed flow-specific medium scheme as well as queue-based traffic observation mechanism and points out the additional contributions associated with the accompanying performance analysis of the proposed scheme. It concludes with the contributions from the energy-efficiency research are and the significance of the capstone application for networked satellite medium access.

This research identifies and characterizes multiple and distinct flows within a wireless sensor network and introduces the term *cooperative wireless sensor network* [18]. In a cooperative wireless sensor network, sensors exchange information to coordinate efforts and maximize application-related performance. This work specifically identifies a high-demand, loss-tolerant flow associated with the sensor data traffic and a low-demand, loss-intolerant, delay-sensitive flow associated with the sensor control traffic. We assert, therefore, that a medium access solution for wireless sensor networks should provide a high throughput medium access service to support the former and a reliable service with minimum end-to-end delay to support the latter.

This dissertation proposes a groundbreaking medium access technique that provides medium access on a per flow basis rather than in aggregation [168]. It introduces the concept of *traffic-adaptive, flow-specific medium access*, a medium access approach that is capable of concurrently providing different medium access service to different traffic flows (i.e., on a per flow basis) and dynamically switching flows between multiple medium access service types to respond to traffic variations. This work formally proves that, given the selection of a suitable switching point, traffic-adaptive, flow-specific medium access is capable of providing better delay performance than contention, non-contention, and hybrid approaches.

This dissertation proposes traffic-adaptive Cooperative Wireless Sensor Medium Access Control (CWS-MAC) [18], [168], a novel traffic-adaptive, flow-specific medium access scheme capable of providing contention- or non-contention-based medium access service on a per flow basis. It shows that traffic-adaptive CWS-MAC, which includes both a novel flow-specific medium access mechanism and a queue-based approach to traffic estimation, outperforms both slotted, non-persistent CSMA (upon which the IEEE 802.11 standard is based) and TDMA.

This research develops a general model for traffic-adaptive, flow-specific medium access control and shows that traditional contention, non-contention, and hybrid medium access schemes are special cases of this model [168]. It examines the two-flow and single-flow cases in detail and develops mean throughput and delay expressions [168],[185].

This research effort develops the first published throughput and delay performance analysis for slotted ALOHA with periodic server vacations [167], which is a contention-based medium access scheme whose service is governed by a fixed cycle composed of alternating active and inactive periods. This type of medium access approach is representative of an energy-efficient, contention-based medium access solution that implements sleep cycles.

This dissertation proposes an energy-efficient version of traffic-adaptive CWS-MAC that employs an adaptive sleep cycle coordinated through the use of preamble sampling [187]. It shows that this novel energy-efficient scheme is capable of achieving low duty cycles while taking advantage of the performance improvements available through a flow-specific medium access approach. A preamble sampling probability parameter is introduced to manage the trade-off between energy efficiency and network throughput and delay performance.

To our knowledge, this work also provides the first comprehensive energy efficiency comparison between centralized and distributed solutions in wireless networks [186]. The analysis includes energy consumption in the transmit, receive, idle and sleep states. It provides a framework comprised of both total and per node expressions that can be applied to wireless sensor network applications such as the beamforming class of unattended battlefield monitoring solutions. It shows that a performance threshold exists between these approaches, which can be exploited through the use of preamble sampling. These results can be extended to any collaborative wireless sensor network.

Finally, this research effort proposes a flow-specific medium access technique to accommodate the large and variable propagation delays and dynamic traffic requirements in a networked satellite system [188],[189]. It quantifies the effect of the large propagation delays on both contention and contention-free satellite-based medium access and shows that the novel flow-specific medium access solution for networked satellite systems provides improved delay performance over both CSMA-based and TDMA-based solutions.

C. FUTURE RESEARCH

Although this work represents a significant contribution to the body of research into medium access for wireless sensor networks and wireless networks in general, there remain a number of open questions that come from this effort. This section identifies a number of these and offers them as interesting topics for future research.

The contention mode of the proposed medium access scheme is based on slotted ALOHA with periodic server vacations. Although certainly a commonly used technique,

other contention-based schemes have been shown to outperform slotted ALOHA. Further research could examine other contention-based implementations, including a slotted non-persistent CSMA version, to further improve the delay and throughput performance in this mode.

To support the performance analysis of the proposed scheme, this research effort develops a general traffic-adaptive, flow-specific model but examines only the two-mode and single-mode cases in detail. Future research could address the general *f*-flow, *m*-mode case and develop the underlying Markov process and associated mapping to arrive at general expressions for mean delay and mean throughput. Additional research could also seek to optimize the choice of the accompanying flow and mode-specific threshold values.

The energy analysis of the proposed energy-efficient scheme focuses on duty cycle analysis. While the duty cycle is certainly representative of energy consumption, it is not equivalent to energy consumption and does not reflect the multiple consumption rates present in the active state. Future research efforts could include an energy consumption analysis based on the time spent and energy expended in the transmit, receive, idle and sleep states. Further research could also explore the energy cost and frequency of transitions between these states as well as the optimum value for the preamble sampling parameter.

Although the proposed medium access scheme does support mobility, it is not explicitly addressed in this work. Further research could analyze the effect of mobility on the delay and throughput performance as well as the energy efficiency of the scheme.

The performance analysis included here assumes that the traffic is Poisson-distributed. This is a reasonable assumption given the results of the work that has been done to date with respect to traffic characterization in sensor networks. However, as discussed in Chapter II, the results to date are not fully conclusive and future research into flow-specific medium access can challenge this assumption and explore the impact of self-similar traffic on the performance of the proposed scheme.

In this work, the proposed wireless medium access scheme relies on a distributed medium access control algorithm. To date, a formal analysis of the algorithm to verify the properties of safety, liveness, and fairness has not been completed and should be a topic of future research.

Finally, the proposed scheme has been modeled in OPNET®, but not yet fielded in an operational testbed. In future research, the scheme could be fielded and a comparison conducted against several other fielded protocols including IEEE 802.11, IEEE 802.15.4, S-MAC, and Z-MAC.

LIST OF REFERENCES

[1] P. Vincent, M. Tummala, and J. McEachen, "A New Method for Distributing Power Usage across a Sensor Network," *Proc. of 3rd Ann. IEEE Comm. Society on Sensor and Ad Hoc Comm. and Networks*, Sep. 2006, vol. 2, pp. 518–526.

[2] M. Batson, and J. McEachen, "A Method for Fast Radio Frequency Direction Finding Using Wireless Sensor Networks," *Proc. of 41st Ann. Hawaii Inter. Conf. on System Sciences*, Hawaii, Jan. 7-10, 2008, pp.495–495.

[3] W. A. Lintz and J. C. McEachen, "A Method for Emphasizing Signal Detection in Wireless Sensor Network Radio Frequency Array Operation," *Proc. of 41st Ann. Hawaii Inter. Conf. on System Sciences*, Hawaii, Jan. 5–8, 2009.

[4] P. Vincent, M. Tummala and J. McEachen, "Connectivity in Sensor Networks," *Proc. of 40th Ann. Hawaii Inter. Conf. on System Sciences*, Hawaii, Jan. 2007.

[5] I. F. Akyildiz and I. H. Kasimoglu, "Wireless sensor and actor networks: Research challenges," *Ad Hoc Networks Journal (Elsevier)*, vol. 2, no. 4, pp. 351–367, Oct. 2004.

[6] I. F. Akyildiz, T. Melodia and K. R. Chowdhury, "A survey on wireless multimedia sensor networks," *Comput. Netw.*, vol. 51, no. 4, pp. 921–960, Mar. 2007.

[7] E. Fishler et al., "MIMO radar: an idea whose time has come," *Proc. of the IEEE Radar Conf.*, Apr. 26-29, 2004, pp. 71–78.

[8] A. Fletcher and F. Robey, "Performance bounds for adaptive coherence of sparse array radar," *Proc. of the Adaptive Sensor Array Processing Workshop*, Lexington, MA, Mar. 2003.

[9] D. J. Rabideau and P. Parker, "Ubiquitous MIMO multifunction digital array radar," *Conf. Rec. of the 37th Asilomar Conference on Signals, Systems and Computers*, Nov. 9–12, 2003, pp. 1057–1064.

[10] D. Gesbert et al., "From theory to practice: an overview of MIMO space-time coded wireless systems," *IEEE Journal on Selected Areas in Communications*, vol. 21, no. 3, pp. 281–302, Apr. 2003.

[11] E. Fishler et al., "Spatial diversity in radars-models and detection performance," *IEEE Trans. on Signal Processing*, vol. 54, no. 3, pp. 823–838, Mar. 2006.

[12] H. Ochiai et al., "Collaborative Beamforming for Distributed Wireless Ad Hoc Sensor Networks," *IEEE Trans. on Signal Processing*, Vol. 53, No. 11, pp. 4110–4124, Nov. 2005.

[13] M. Tummala, C. C. Wai and P. Vincent, "Distributed Beamforming in Wireless Sensor Networks," *Conf. Rec. of the 39th Asilomar Conf. on Signals, Systems and Computers*, Oct. 28 – Nov. 1, 2005, pp. 793–797.

[14] T. O. Walker, M. Tummala, and J. B. Michael, "Pulse transmission scheduling for a distributed system of cooperative radars," *Proc. 2006 IEEE/SMC Int. Conf. on System of Systems Engineering*, Los Angeles, CA, Apr. 2006, no. 6, pp. 24–26.

[15] A. L. Hume and C. J. Baker, "Netted Radar Sensing," *Proc. CIE Int. Conf. on Radar, IEEE*, Beijing, China, Oct. 2001, pp. 110–114.

[16] I. F. Akyildiz, et al., "A survey on sensor networks," *IEEE Communications Magazine*, vol. 40, no. 8, pp. 102–114, Aug. 2002.

[17] D. Niculescu, "Communication paradigms for sensor networks," *IEEE Communications Magazine*, vol. 43, no. 3, pp. 116–122, Mar. 2005.

[18] T. O. Walker, M. Tummala, and J. McEachen, "Distributed, flow-based priority medium access control for cooperative multihop wireless Sensor networks," *Proc. of the 41st Ann. Hawaii Intl. Conf. on System Sciences*, Hawaii, Jan. 2008.

[19] V. Rajendran, K. Obraczka, and J. J. Garcia-Luna-Aceves, "Energy-efficient, collision-free medium access control for wireless sensor networks," *Wirel. Netw.*, vol. 12, no. 1, pp. 63–78, Feb. 2006.

[20] N. Abramson, "The ALOHA System – Another Alternative for Computer Communications," *Proc. of the AFIPS Fall Joint Computer Conf.*, 1970, vol. 37, pp. 281–285.

[21] L. Kleinrock and F. Tobagi, "Packet Switching in Radio Channels: Part I—Carrier Sense Multiple-Access Modes and Their Throughput-Delay Characteristics," *IEEE Trans. on Communications*, vol. 23, no. 12, pp. 1400–1416, Dec. 1975.

[22] W. Ye, J. Heidemann, and D. Estrin, "Medium Access Control with Coordinated Adaptive Sleeping for Wireless Sensor Networks," *IEEE/ACM Trans. Net.*, vol. 12, no. 3, pp. 493–506, Jun. 2004.

[23] I. Rhee et al., "Z-MAC: a hybrid MAC for wireless sensor networks," *Proc. of the 3rd Intl. Conf. on Embedded Networked Sensor Systems*, San Diego, California, Nov. 2-4, 2005, pp. 90–101.

[24] "IEEE Standard for Information technology – Telecommunications and information exchange between systems – Local and metropolitan area networks – Specific requirements – Part 11: Wireless LAN Medium Access Control (MAC) and Physical Layer (PHY) Specifications," *IEEE Std 802.11-2007 (Revision of IEEE Std 802.11-1999)*, Jun. 12, 2007.

[25] "IEEE Standard for Information technology – Telecommunications and information exchange between systems – Local and metropolitan area networks – Specific requirements Part 15.4: Wireless Medium Access Control (MAC) and Physical Layer (PHY) Specifications for Low-Rate Wireless Personal Area Networks (WPANs)," *IEEE Std 802.15.4-2006 (Revision of IEEE Std 802.15.4-2003)*, 2006.

[26] S. Lam, "Delay analysis of a time division multiple Access (TDMA) channel," *IEEE Trans. on Commun.*, vol. 25, no. 12, pp. 1489–1494, Dec. 1977.

[27] V. Rajendran, K. Obraczka, and J.J. Garcia-Luna-Aceves, "Energy-efficient, collision-free medium access control for wireless sensor networks," *Wireless. Networks*, vol. 12, no. 1, pp. 63–78, Feb. 2006.

[28] I. Rhee, A. Warrier, M. Aia, and J. Min, "Z-MAC: A hybrid MAC for wireless sensor networks," *Proc. of the 3rd Int.l Conf. on Embedded Networked Sensor Systems*, San Diego, California, Nov. 2005, pp. 90–101.

[29] P. Vincent, "Energy conservation in wireless sensor networks," Ph.D. dissertation, Dept. of Elec. and Comp. Engr., Naval Postgraduate School, Monterey, CA, 2007.

[30] I. Demirkol, C. Ersoy, and F. Alagoz, "MAC protocols for wireless sensor networks: a survey," *IEEE Communications Magazine*, vol. 44, no. 4, pp. 115–121, Apr. 2006.

[31] M. Athanassoulis, I. Alagiannis, and S. Hadjiefthymiades, "Energy Efficiency in Wireless Sensor Networks: A Utility-Based Architecture," *European Wireless 2007*, Paris, France, Apr. 1–4, 2007.

[32] D. Culler, D. Estrin, and M. Srivastava, "An overview of sensor networks," *IEEE Computer*, pp. 41–49, Aug. 2004.

[33] J. N. Al-Karaki and A. E. Kamal, "Routing techniques in wireless sensor networks: a survey," *IEEE Wireless Communications*, vol. 11, no. 6, pp. 6–28, Dec. 2004.

[34] "TelosB data sheet," Crossbow Technology. Available: http://www.xbow.com/Products/Product_pdf_files/Wireless_pdf/TelosB_Datasheet.pdf, accessed Sep. 10, 2009.

[35] "SunSPOT Theory of Operation," SunSPOT World. Available: http://www.sunspotworld.com/docs/Red/SunSPOT-TheoryOfOperation.pdf, accessed Sep. 10, 2009.

[36] "Mica2 data sheet," Crossbow Technology. Available: http://www.xbow.com/Products/Product_pdf_files/Wireless_pdf/MICA2_Datasheet.pdf, accessed Sep. 10, 2009.

[37] "MicaZ data sheet," Crossbow Technology. Available: http://www.xbow.com/Products/Product_pdf_files/Wireless_pdf/MICAz_Datasheet.pdf, accessed Sep. 10, 2009.

[38] J. L. Hill and D. E. Culler, "Mica: a wireless platform for deeply embedded networks," *IEEE Micro*, vol. 22, no .6, pp. 12–24, Nov/Dec. 2002.

[39] A. P. Petropulu and X. Yang, "Data Traffic Modeling: A Signal Processing Perspective," in *Nonlinear Signal and Image Processing: Theory, Methods, and Applications*, eds. K.E. Barner and G. Arce, CRC Press, 2003.

[40] T. G. Robertazzi, *Computer Networks and Systems: Queueing Theory and Performance Evaluation (Third Edition)*. New York: Springer, 2000.

[41] W. Stallings, *High-Speed Networks and Internets: Performance and Quality of Service, Second Edition*. Upper Saddle River, NJ: Prentice Hall, 2002.

[42] K. Park and W. Willinger, *Self-similar Network Traffic and Performance Evaluation*. New York: John Wiley & Sons, 2000.

[43] W. E. Leland et al., "On the self-similar nature of Ethernet traffic (extended version)," *IEEE/ACM Trans. on Networking*, vol. 2, no. 1, pp. 1–15, Feb. 1994.

[44] B. Tsybakov and N. D. Georganas, "On self-similar traffic in ATM queues: definitions, overflow probability bound, and cell delay distribution," *IEEE/ACM Trans. on Networking*, vol. 5, no. 3, pp. 397–409, Jun. 1997.

[45] S. Song; J. K.-Y. Ng, and B. Tang, "Some results on the self-similarity property in communication networks," *IEEE Trans. on Communications*, vol.5 2, no. 10, pp. 1636–1642, Oct. 2004.

[46] J. C. McEachen and W. W. Beng, "Characterization of traffic in wireless sensor networks," *15th IEEE Intl. Conf. on Networks*, Nov. 19-21, 2007, pp. 437–442.

[47] Q. Liang, "Ad hoc wireless network traffic-self-similarity and forecasting," *IEEE Communications Letters*, vol. 6, no. 7, pp. 297-299, Jul. 2002.

[48] L. Kleinrock, *Queueing Systems, Volume I: Theory*. New York: John Wiley & Sons, 1975.

[49] A. Leon-Garci, Alberto, *Probability and Random Processes for Electrical Engineering*. Reading, Massachusetts: Addison-Wesley Publishing Company, 1989.

[50] I. Norros, "A Storage Model with Self-Similar Input," *Queueing Syst.*, vol. 16, pp. 387–396, 1994.

[51] B. Hull, K. Jamieson, and H. Balakrishnan, "Mitigating congestion in wireless sensor networks", Proc. of the 2nd Intl. Conf. on Embedded Networked Sensor Systems, Baltimore, MD, Nov. 3-5, 2004, pp. 134–147.

[52] M. Zawodniok and S. Jagannathan, "Predictive Congestion Control Protocol for Wireless Sensor Networks," *IEEE Trans. on Wireless Communications*, vol. 6, no. 11, pp. 3955–3963, Nov. 2007

[53] C. Wan, S. B. Eisenman, and A. T. Campbell, "CODA: congestion detection and avoidance in sensor networks," *Proc. of the 1st Intl. Conf. on Embedded Networked Sensor Systems*, Los Angeles, California, Nov. 5-7, 2003, pp. 266–279.

[54] S. Rangwala et al., "Interference-aware fair rate control in wireless sensor networks," *Proc. of the 2006 Conf. on Applications, Technologies, Architectures, and Protocols for Computer Communications*, Pisa, Italy, Sep. 11-15, 2006, pp. 63–74.

[55] C. Wang et al., “Priority-based Congestion Control in Wireless Sensor Networks,” *IEEE Intl. Conf. on Sensor Networks, Ubiquitous, and Trustworthy Computing*, vol. 1, Jun. 5–7, 2006, pp. 22–31.

[56] T. Holmberg et al., “Traffic Estimation in Mobile TDMA-based Ad Hoc Networks,” *Proc. 6th Ann. Mediterranean Ad Hoc Networking Workshop*, Corfu, Greece, Jun. 12–15, 2007.

[57] K. Karenos, V. Kalogeraki, and S. V. Krishnamurthy, “Cluster-based congestion control for sensor networks,” *ACM Trans. Sensor Networks*, vol. 4, no. 1, pp. 1–39, Jan. 2008.

[58] P. M. Shankar, *Introduction to Wireless Systems*. New York: John Wiley & Sons, 2002.

[59] T. S. Rappaport, *Wireless Communications: Principles and Practice (Second Edition)*. Upper Saddle River, NJ: 2002.

[60] G. Zhou,et al., “Impact of radio irregularity on wireless sensor networks,” *Proc. of the 2nd Intl.Conf. on Mobile Systems, Applications, and Services*, Boston, MA, Jun. 6–9, 2004, pp. 125–138.

[61] W. Stallings, *Wireless Communications and Nettworks (Second Edition)*. Upper Saddle River, NJ: Pearson Prentice Hall, 2005.

[62] B. Krishnamachari, *Networking Wireless Sensors*. Cambridge: Cambridge University Press, 2005.

[63] P. Gupta and P. R. Kumar, “The capacity of wireless networks,” *IEEE Trans. on Information Theory*, vol. IT-46, pp. 388–404, Mar. 2000.

[64] L. G. Roberts, “ALOHA packet system with and without slots and capture,” *Computer Communication Review*, vol. 5, no. 2, pp. 28–42, 1972.

[65] J. Polastre, J. Hill, and D. Culler, “Versatile low power media access for wireless sensor networks,” *Proc. of the 2nd Intl. Conf. on Embedded Networked Sensor Systems*, Baltimore, MD, Nov. 3–5, 2004, pp. 95–107.

[66] F. Tobagi and L. Kleinrock, “Packet Switching in Radio Channels: Part II—The Hidden Terminal Problem in Carrier Sense Multiple-Access and the Busy-Tone Solution,” *IEEE Trans. on Communications*, vol. 23, no. 12, pp. 1417–1433, Dec. 1975.

[67] P. Karn, “MACA—A new channel access method for packet radio,” *Proc. 9th ARRL Comput. Netw. Conf.*, 1990, pp. 134–140.

[68] I. Rhee et al., “DRAND: Distributed Randomized TDMA Scheduling For Wireless Ad-hoc Networks,” *Proc. of the 7th ACM Intl. Symp. on Mobile Ad Hoc Networking and Computing*, Florence, Italy, May 22–25, 2006, pp. 190–201.

[69] A. Ephremides, and T. V. Truong, “Scheduling broadcasts in multihop radio networks,” *IEEE Trans. on Communications*, vol. 38, no. 4, pp. 456–460, Apr. 1990.

[70] S. Ramanathan, "A unified framework and algorithm for (T/F/C)DMA channel assignment in wireless networks," *Proc. 16th Ann. Joint Conf. of the IEEE Computer and Communications Societies*, Apr. 7–12, 1997, pp.900–907.

[71] C. Zhu and M. S. Corson, "A five-phase reservation protocol (FPRP) for mobile ad hoc networks ," *Proc. 17th Ann. Joint Conf. of the IEEE Computer and Communications Societies*, Mar. 29 –Apr. 2, 1998, pp.322–331.

[72] R. Gandhi and S. Parthasarathy, "Distributed Algorithms for Connected Domination in Wireless Networks," *Journal of Parallel and Distributed Computing*, vol. 67, no. 7, pp. 848–862, Jul. 2007.

[73] L. Bao and J. J. Garcia-Luna-Aceves, "A New Approach to Channel Access Scheduling for Ad Hoc Networks," *7th Ann. Int'l. Conf. Mobile Comp. and Net.*, 2001, pp. 210–21.

[74] V. Rajendran, J. J. Garcia-Luna-Aveces and K. Obraczka, "Energy-efficient, application-aware medium access for sensor networks," *IEEE Intl. Conf. on Mobile Adhoc and Sensor Systems*, Nov. 7–10, 2005, pp. 1–8.

[75] A. Boukerche and D. Turgut, "Secure time synchronization protocols for wireless sensor networks," *IEEE Wireless Communications*, vol. 14, no. 5, pp. 64–69, Oct. 2007.

[76] K. Romer, P. Blum, and L. Meier, "Time synchronization and calibration in wireless sensor networks," in *Handbook of Sensor Networks: Algorithms and Architectures*, I. Stojmenovic, Ed. John Wiley & Sons, 2005, pp. 199–237.

[77] Qun Li and D. Rus, "Global clock synchronization in sensor networks," *IEEE Trans. on Computers*, vol. 55, no. 2, pp. 214–226, Feb. 2006.

[78] F. Sivrikaya and B. Yener, "Time synchronization in sensor networks: a survey," *IEEE Network*, vol. 18, no. 4, pp. 45–50, Jul.–Aug. 2004.

[79] M. Maróti et al., "The flooding time synchronization protocol," *Proc. of the 2nd Intl. Conf. on Embedded Networked Sensor Systems*, Baltimore, MD, Nov. 3–5, 2004.

[80] J. Elson, L. Girod, and D. Estrin, "Fine-grained network time synchronization using reference broadcasts," *Proc. of the 5th Symp. on Operating Systems Design and Implementation*, Boston, Massachusetts, Dec. 9–11, 2002.

[81] S. Ganeriwal, R. Kumar, and M. B. Srivastava, "Timing-sync protocol for sensor networks," *Proc. of the 1st Intl. Conf. on Embedded Networked Sensor Systems*, Los Angeles, California, Nov. 5–7, 2003, pp. 138–149.

[82] J. Elson and D. Estrin, "Time synchronization for wireless sensor networks," *Proc. 15th Intl. Parallel and Distributed Processing Symp.*, Apr. 2001, pp. 1965–1970.

[83] D. L. Mills, "Internet time synchronization: the network time protocol," *IEEE Trans. on Communications*, vol. 39, no. 10, pp. 1482–1493, Oct. 1991.

[84] *NAVSTAR GPS User Equipment Introduction*, Sep. 1996, Publications library, Navigation Center, United States Coast Guard. Available: http://www.navcen.uscg.gov/pubs/gps/gpsuser/gpsuser.pdf, accessed Sep. 10, 2009.

[85] R. Jurdak, C. V. Lopes, and P. Baldi, "A Survey, Classification and Comparative Analysis of Medium Access Protocols for Ad Hoc Networks," *IEEE Communications Surveys*, vol. 6, no. 1, pp. 2–16, 2004.

[86] D. Son, B. Krishnamachari, and J. Heidemann, „Experimental study of concurrent transmission in wireless sensor networks," *Proc. of the 4th Intl. Conf. on Embedded Networked Sensor Systems*, Boulder, Colorado, Oct. 31 – Nov. 3, 2006, pp. 237–250.

[87] K. Whitehouse et al., "Exploiting the Capture Effect for Collision Detection and Recovery," *2nd IEEE Workshop on Embedded Networked Sensors*, May 30–31, 2005, pp. 45–52.

[88] W. R. Heinzelman, A. Chandrakasan, and H. Balakrishnan, "Energy-efficient communication protocol for wireless microsensor networks," *Proc. of the 33rd Ann. Hawaii Intl. Conf. on System Science*, Jan. 4–7, 2000, pp. 2–10.

[89] W. B. Heinzelman, A. P. Chandrakasan and H. Balakrishnan, "An application-specific protocol architecture for wireless microsensor networks," *IEEE Trans. on Wireless Communications*, vol. 1, no. 4, pp. 660–670, Oct. 2002.

[90] R. Min and A. Chandrakasan, "MobiCom poster: top five myths about the energy consumption of wireless communication," *SIGMOBILE Mob. Comput. Commun. Rev.* 7, Jan. 2003, pp. 65–67.

[91] A. Y. Wang and C. G. Sodini, "A simple energy model for wireless microsensor transceivers," *IEEE Global Telecommunications Conference*, Nov. 29–Dec. 3, 2004, vol. 5, pp. 3205–3209.

[92] Q. Wang, M. Hempstead, and W. Yang, "A Realistic Power Consumption Model for Wireless Sensor Network Devices," *3rd Ann. IEEE Communications Society on Sensor and Ad Hoc Communications and Networks*, Sep. 2006, pp. 286–295.

[93] R, Jurdak, A. G. Ruzzelli, and G. O'Hare, "Adaptive Radio Modes in Sensor Networks: How Deep to Sleep?" *5th Ann. IEEE Communications Society on Sensor and Ad Hoc Communications and Networks*, San Francisco, CA, Jun. 16–20, 2008.

[94] J. Polastre, R. Szewczyk, and D. Culler, "Telos: enabling ultra-low power wireless research," *4th Intl. Symp. on Information Processing in Sensor Networks*, Apr. 15, 2005, pp. 364–369.

[95] Y.-C. Tseng, C.-S. Hsu and T.-Y. Hsieh, "Power-saving protocols for IEEE 802.11-based multi-hop ad hoc networks," *21st Proc. Ann. Joint Conf. of the IEEE Computer and Communications Societies*, 2002, pp. 200–209.

[96] S. Singh, and C. S. Raghavendra, “PAMAS—power aware multi-access protocol with signaling for ad hoc networks,” *SIGCOMM Comput. Commun. Rev*., vol. 28, no. 3, pp. 5–26, Jul. 1998.

[97] D. Culler, “Reuniting Wireless Sensor Networks with the IP Architecture (Keynote),” *5th Ann. IEEE Communications Society on Sensor and Ad Hoc Communications and Networks*, San Francisco, CA, Jun. 16–20, 2008.

[98] J. M Rabaey et al., “PicoRadio supports ad hoc ultra-low power wireless networking,” *IEEE Computer*, vol. 33, no. 7, pp. 42–48, Jul. 2000.

[99] E. Shih, P. Bahl, and M. J. Sinclair, “Wake on wireless: an event driven energy saving strategy for battery operated devices,” *Proc. of the 8th Annual Intl. Conf. on Mobile Computing and Networking*, Atlanta, Georgia, Sep. 23–28, 2002, pp. 160–171.

[100] A. El-Hoiydi, “ALOHA with preamble sampling for sporadic traffic in ad hoc wireless sensor networks,” *IEEE Intl. Conf. on Communication*, vol. 5, Apr. 2002, pp. 3418–3423.

[101] A. El-Hoiydi and J. D. Decotignie, “WiseMAC: An ultra low-power MAC protocol for multi-hop wireless sensor networks,” *1st Intl. Workshop on Algorithmic Aspects of Wireless Sensor Networks*, Turku, Finland, Jul. 16, 2004.

[102] T. V. Dam and K. Langendoen, “An Adaptive Energy-Efficient MAC Protocol for Wireless Sensor Networks,” *1st ACM Conf. Embedded Networked Sensor Sys*., Los Angeles, CA, Nov. 2003.

[103] G. Lu, B. Krishnamachari, and C. S. Raghavendra, “An Adaptive Energy-Efficient and Low-Latency MAC for Data Gathering in Wireless Sensor Networks,” Proc. 18th Int’l. Parallel and Distrib. Processing Symp., Apr. 2004.

[104] Wei Wang et al., “An Energy Efficient Pre-Schedule Scheme for Hybrid CSMA/TDMA MAC in Wireless Sensor Networks,” *10th IEEE Singapore Intl. Conf. on Communication Systems*, Oct. 2006, pp. 1–5.

[105] V. Kawadia and P. R. Kumar, “Principles and protocols for power control in wireless ad hoc networks,” *IEEE Journal on Selected Areas in Communications*, vol. 23, no. 1, pp. 76–88, Jan. 2005.

[106] A. Muqattash and M. Krunz, “POWMAC: a single-channel power-control protocol for throughput enhancement in wireless ad hoc networks,” *IEEE Journal on Selected Areas in Communications*, vol. 23, no. 5, pp. 1067–1084, May 2005.

[107] A. Muqattash, M Krunz, and S-J Lee, “A Perspective on the Design of Power Control for Mobile Ad Hoc Networks,” *Mobile, Wireless, and Sensor Networks: Technology, Applications and Future Directions*, ed. R. Shorey et al., John Wiley & Sons, 2006, Ch. 4.

[108] M. Krunz, A. Muqattash and S.-J. Lee, “Transmission power control in wireless ad hoc networks: challenges, solutions and open issues,” *IEEE Network*, vol. 18, no. 5, pp. 8–14, Sep.–Oct. 2004.

[109] E.-S. Jung and N. H. Vaidya, "A power control MAC protocol for ad-hoc networks," *ACM MOBICOM*, 2002.

[110] J. P. Monks, V. Bharghavan, and W.-M. W. Hwu, "A power controlled multiple access protocol for wireless packet networks," *Proc. 20th Ann. Joint Conference of the IEEE Computer and Communications Societies*, 2001, pp. 219–228.

[111] S.-L. Wu; Y.-C. Tseng and J.-P. Sheu, "Intelligent medium access for mobile ad hoc networks with busy tones and power control," *IEEE Journal on Selected Areas in Communications*, vol. 18, no. 9, pp.1 647–1657, Sep. 2000.

[112] A. Muqattash and M. Krunz, "Power controlled dual channel (PCDC) medium access protocol for wireless ad hoc networks," *IEEE 22nd Ann. Joint Conf. of the IEEE Computer and Communications Societies*, Mar. 30–Apr. 3, 2003, pp. 470–480.

[113] I. F. Akyildiz, and X. Wang, "Cross-Layer Design in Wireless Mesh Networks," *IEEE Trans. on Vehicular Technology*, vol. 57, no. 2, pp. 1061–1076, Mar. 2008.

[114] M. van Der Schaar and N. Sai Shankar, "Cross-layer wireless multimedia transmission: challenges, principles, and new paradigms," *IEEE Wireless Communications*, vol. 12, no. 4, pp. 50–58, Aug. 2005.

[115] V. Kawadia and P. R. Kumar, "A cautionary perspective on cross-layer design," *IEEE Wireless Communications*, vol. 12, no. 1, pp. 3–11, Feb. 2005.

[116] T. ElBatt and A. Ephremides, "Joint scheduling and power control for wireless ad hoc networks," *IEEE Trans. on Wireless Communications*, vol. 3, no. 1, pp. 74–85, Jan. 2004.

[117] U. C. Kozat, I. Koutsopoulos, and L. Tassiulas, "Cross-Layer Design for Power Efficiency and QoS Provisioning in Multi-Hop Wireless Networks," *IEEE Trans. on Wireless Communications*, vol. 5, no. 11, pp. 3306–3315, Nov. 2006.

[118] W. L. Huang and K. B. Letaief, "Cross-Layer Scheduling and Power Control Combined With Adaptive Modulation for Wireless Ad Hoc Networks," *IEEE Trans. on Communications*, vol. 55, no. 4, pp. 728–739, Apr. 2007.

[119] L. Van Hoesel et al., "Prolonging the lifetime of wireless sensor networks by cross-layer interaction," *IEEE Wireless Communications*, vol. 11, no. 6, pp. 78–86, Dec. 2004.

[120] M. L. Sichitiu, "Cross-layer scheduling for power efficiency in wireless sensor networks," *23rd Ann. Joint Conf. of the IEEE Computer and Communications Societies*, vol. 3, Mar. 7–11, 2004, pp. 1740–1750.

[121] Y. E. Sagduyu and A. Ephremides, "On Joint MAC and Network Coding in Wireless Ad Hoc Networks," *IEEE Trans. on Information Theory*, vol. 53, no. 10, pp. 3697–3713, Oct. 2007.

[122] X. Lin, N. B. Shroff, and R. Srikant, "A Tutorial on Cross-Layer Optimization in Wireless Networks," *IEEE Journal on Selected Areas in Communications* vol. 24, no. 8, pp. 1452–1463, Aug. 2006.

[123] M. Chiang, "Balancing transport and physical layers in wireless multihop networks: jointly optimal congestion control and power control," *IEEE Journal on Selected Areas in Communications*, vol. 23, no. 1, pp. 104–116, Jan. 2005.

[124] S. Cui et al., "Cross-Layer Energy and Delay Optimization in Small-Scale Sensor Networks," *IEEE Trans. on Wireless Communications*, vol. 6, no. 10, pp. 3688–3699, Oct. 2007.

[125] T. Melodia, M. C. Vuran, and D. Pompili, "The State of the Art in Cross-layer Design for Wireless Sensor Networks," *Springer Lecture Notes in Computer Science (LNCS)*, pp. 78–92, 2006.

[126] W. Su, and T. L. Lim, "Cross-Layer Design and Optimization for Wireless Sensor Networks," *7th ACIS Intl. Conf. on Software Engineering, Artificial Intelligence, Networking, and Parallel/Distributed Computin*, Jun. 19–20, 2006, pp. 278–284.

[127] M. Chiang et al., "Layering as Optimization Decomposition: A Mathematical Theory of Network Architectures," *Proceedings of the IEEE*, vol. 95, no. 1, pp. 255–312, Jan. 2007.

[128] V. Bharghavan, et al., "MACAW: a media access protocol for wireless LAN's," *Proc. of the Conf. on Communications Architectures, Protocols and Applications*, London, United Kingdom, Aug. 31 – Sep. 2, 1994, pp. 212–225.

[129] C. L. Fullmer, and J. J. Garcia-Luna-Aceves, "Floor acquisition multiple access (FAMA) for packet-radio networks," *Proc. of the Conf. on Applications, Technologies, Architectures, and Protocols For Computer Communication*, Cambridge, Massachusetts, Aug. 28 – Sep. 1, 1995, pp. 262–273.

[130] F. Talucci, and M. Gerla, "MACA-BI (MACA by invitation). A wireless MAC protocol for high speed ad hoc networking," *1997 IEEE 6th Intl. Conf. on Universal Personal Communications Record*, Oct 12–16, 1997, vol. 2, pp. 913–917.

[131] C.-K. Toh et al., "MARCH: a medium access control protocol for multihop wireless ad hoc networks," *Proc. 21st Century Military Communications Conf.*, 2000, pp. 512–516.

[132] K. Kredo and P. Mohapatra, "Medium access control in wireless sensor networks," *Comput. Netw.*, vol. 51, no. 4, pp. 961–994, Mar. 2007.

[133] S. Xu, and T. Saadawi, "Does the IEEE 802.11 MAC protocol work well in multihop wireless ad hoc networks?" *IEEE Communications Magazine*, vol. 39, no. 6, pp. 130–137, Jun. 2001.

[134] Y. E. Sagduyu and A. Ephremides, "The problem of medium access control in wireless sensor networks," *IEEE Wireless Communications*, vol. 11, no. 6, pp. 44–53, Dec. 2004.

[135] "IEEE Standard for Information Technology – Telecommunications and information exchange between systems – Local and metropolitan area networks – specific requirement Part 15.4: Wireless Medium Access Control (MAC) and Physical Layer (PHY) Specifications for Low-Rate Wireless Personal Area Networks (WPANs)," *IEEE Std 802.15.4a-2007 (Amendment to IEEE Std 802.15.4-2006)*, 2007

[136] M. Kohvakkaet al., "Performance analysis of IEEE 802.15.4 and ZigBee for large-scale wireless sensor network applications," *Proc. of the 3rd ACM Intl. Workshop on Performance Evaluation of Wireless Ad Hoc, Sensor and Ubiquitous Networks*, Terromolinos, Spain, Oct. 6, 2006, pp. 48–57.

[137] S. Pollin et al., "MAC 08-5 – Performance Analysis of Slotted Carrier Sense IEEE 802.15.4 Acknowledged Uplink Transmissions," *IEEE Wireless Communications and Networking Conference*, Mar. 31 – Apr. 3, 2008, pp.1559–1564.

[138] L.-C. Ko, Y.-C. Liu and H.-W. Fang, "Design and implementation of IEEE 802.15.4 beacon-enabled network devices," *4th Ann. IEEE Intl. Conf. on Pervasive Computing and Communications*, Mar. 13–17, 2006, pp. 1–5.

[139] S. Pollin et al., "WLC10-5: Performance Analysis of Slotted Carrier Sense IEEE 802.15.4 Medium Access Layer," *IEEE Global Telecommunications Conf.*, Nov. 2006, pp.1–6.

[140] B. Bougard et al., "Energy efficiency of the IEEE 802.15.4 standard in dense wireless microsensor networks: modeling and improvement perspectives," *Proc. Design, Automation and Test in Europe*, Mar. 7–11, 2005, pp. 196–201.

[141] P. Lin, C. Qiao, and X. Wang, "Medium Access Control with a Dynamic Duty Cycle for Sensor Networks," *IEEE WCNC*, vol. 3, Mar. 2004, pp. 1534–1539.

[142] L. F. W. van Hoesel and P. J. M. Havinga. "A Lightweight Medium Access Protocol (LMAC) for Wireless Sensor Networks: Reducing Preamble Transmissions and Transceiver State Switches," *Proc. of the Intl. Conf. on Networked Sensing Systems*, Jun. 2004.

[143] S. Chatterjea, L. F. W. van Hoesel, and P. J. M. Havinga, "AI-LMAC: An adaptive, information-centric and lightweight MAC protocol for wireless sensor networks," in Proc. of the Intelligent Sensors, Sensor Networks, and Information Processing Conf., Dec. 2004, pp. 381–388.

[144] L. F. W. van Hoesel and P. J. M. Havinga, "Collision-free Time Slot Reuse in Multi-hop Wireless Sensor Networks," Proc. of the 2005 Intl. Conf. on Intelligent Sensors, Sensor Networks and Information Processing Conference, Dec. 5–8, 2005, pp. 101–107.

[145] W. L. Lee, A. Datta, and R. Cardell-Oliver, "FlexiTP: A Flexible-Schedule-Based TDMA Protocol for Fault-Tolerant and Energy-Efficient Wireless Sensor Networks," IEEE Trans. on Parallel and Distributed Systems, vol. 19, no. 6, pp. 851–864, Jun. 2008.

[146] S. S. Kulkarni, “TDMA Services for Sensor Networks,” Proc. 24th Int’l. Conf. Distrib. Comp. Sys. Wksps., Mar. 2004, pp. 604–09.

[147] K-J Paek et al., “Priority-Based Medium Access Control Protocol for Providing QoS in Wireless Sensor Networks,” *IEICE Trans. on Information and Systems*, vol. E90-D(9), pp. 1448–1451, Sep. 2007.

[148] S. Parthasarathy and R. Gandhi, “Distributed algorithms for coloring and domination in wireless ad hoc networks,” *Proc. Conf. on Foundations of Software Tech. and Theoretical Comp. Science*, Dec. 2004, pp. 447–459.

[149] S. Floyd and V. Jacobson, “Random early detection gateways for congestion avoidance,” *IEEE/ACM Trans. on Networking,* vol. 1, no. 4, pp. 397–413, Aug. 1993.

[150] H. Stark and J. W. Woods, Probability and Random Processes with Applications to Signal Processing (Third Edition). Upper Saddle River, NJ: Prentice Hall, 2002.

[151] R. F. Gebhard, “A queueing process with bilevel hysteretic service-rate control,” *Naval Res. Log. Quart.*, vol. 14, pp. 55–68, 1967.

[152] M. Yadin and P. Naor, “On queueing systems with variable service capacities,” *Naval Res. Log. Quart.*, vol. 14, pp. 43–54, 1967.

[153] W. C. Chan, *Performance Analysis of Telecommunications and Local Area Networks*, Boston: Kluwer Academic Publishers, 2000.

[154] L. Kleinrock, and S. S. Lam, “Packet switching in a multi-access broadcast channel: performance evaluation,” *IEEE Trans. on Commun.*, vol. 23, no. 4, pp. 410–423, Apr. 1975.

[155] N. Papalexidis, T. O. Walker, C. Gkionis, M. Tummala, and J. McEachen, “A distributed approach to beamforming in a wireless sensor network, *Forty-first Asilomar Conf. on Signals, Systems and Computers*, Monterey, CA, Nov. 2007, pp. 606–610.

[156] N. Papalexidis T. O. Walker, M. Tummala, and J. McEachen, “An energy-efficient and distributed approach to beamforming in a wireless sensor network, *Forty-second Asilomar Conf. on Signals, Systems and Computers*, Monterey, CA, Nov. 2008, pp. 878–881.

[157] M. Cetin et al., “Distributed fusion in sensor networks,” *IEEE Signal Processing Magazine*, vol. 23, no. 4, pp.42–55, Jul. 2006.

[158] J.-J. Xiao, et al., “Distributed compression-estimation using wireless sensor networks,” *IEEE Signal Processing Magazine,* vol. 23, no. 4, pp. 27–41, Jul. 2006.

[159] J. C. Chen, K. Yao, and R. E. Hudson, “Source localization and beamforming,” *IEEE Signal Processing Magazine*, vol. 19, no. 2, pp. 30–39, Mar. 2002.

[160] R. Mudumbai, G. Barriac, and U. Madhow, “On the feasibility of distributed beamforming in wireless networks,” *IEEE Trans. on Wireless Communications*, vol. 6, no. 5, pp. 1754–1763, May 2007.

[161] S. Singh, M. Woo, and C. S. Raghavendra, “Power-aware routing in mobile ad hoc networks,” *Proc. ACM/IEEE Int. Conf. on Mobile Computing and Networking (MOBICOM)*, Dallas, TX, Oct. 1998, pp. 181–190.

[162] N. A. Pantazis, D. D. Vergados, “A survey on power control issues in wireless sensor networks,” *IEEE Communications Surveys & Tutorials*, vol. 9, no. 4, pp. 86–107, 2007.

[163] C. Bettstetter and J. Eberspacher, “Hop distances in homogeneous ad hoc networks,” *Proc. of 57th IEEE Semiannual Vehicular Technology Conf.*, Apr. 22–25, 2003, vol. 4, pp. 2286–2290.

[164] S. Mukherjee and D. Avidor, “Connectivity and transmit-energy considerations between any pair of nodes in a wireless ad hoc network subject to fading,” *IEEE Trans. on Vehicular Technology*, vol. 57, no. 2, pp. 1226–1242, Mar. 2008.

[165] G. Golub and C. F. Van Loan, *Matrix Computations*. Baltimore, MA: The Johns Hopkins University Press, 1996.

[166] S. J. Leon, *Linear Algebra with Applications*. Upper Saddle River, NJ: Prentice Hall, 2006.

[167] T. O. Walker, M. Tummala, and J. McEachen, “Performance analysis of slotted ALOHA with periodic server vacations for energy-efficient medium access,” *Forty-second Asilomar Conf. on Signals, Systems and Computers*, Monterey, CA, Nov. 2009, accepted for publication.

[168] T. O. Walker, M. Tummala, and J. McEachen, “Traffic-adaptive, flow-specific medium access control for wireless networks,” *Proc. of the IEEE Wireless Communications and Networking Conf.*, Budapest, Hungary, Apr. 5–8, 2009.

[169] G. Katsis, T. O. Walker, P. Katopodis, M. Tummala and J. B. Michael, “ Multistage Security for Hybrid, Large-scale, Mobile Wireless Sensor Networks for Missile Defense,” *Proc. 2008 IEEE Int. Conf. on System of Systems Engineering*, Monterey, CA, Jun. 2008, pp. 1–6.

[170] P. Katopodis, G. Katsis, T. O. Walker, M. Tummala, and J. B. Michael, “A Hybrid, Large-scale Wireless Sensor Network for Missile Defense,” Proc. 2007 IEEE Int. Conf. on System of Systems Engineering, San Antonio, TX, Apr. 2007, pp. 1–5.

[171] Pratt, et. al., “An Operational and Performance Overview of the Iridium Low Earth Orbit Satellite System,” *IEEE Comm. Surveys*, Second Qrtr. 1999.

[172] E. J. Knoblock, et al., “Network Configuration Analysis for Formation Flying Satellites,” *Proc. IEEE Aerospace Conf.*, Mar. 10–17, 2001, pp. 991–1000.

[173] E. C. Megla, V. Konangi, and M. A. Seibert, "Protocols for Inter-Satellite Communications in a Formation Flying System," *20th AIAA Int'l. Communication Satellite Systems Conf. and Exhibit*, Montreal, Quebec, Canada, May 12–15, 2002.

[174] T. Vladimirova, et al., "Enabling Technologies for Distributed Picosatellite Missions in LEO." *Proc. of the NASA/ESA Conf. on Adaptive Hardware and Systems*, 2006.

[175] K. Sidibeh and T. Vladimirova, "IEEE802.11 Optimisation Techniques for Inter-Satellite Links in LEO Networks," *Int. Conf. on Advanced Comm. Technology*, Feb 20-22, 2006, pp. 1177–1182.

[176] K. Sidibeh and T. Vladimirova, "Wireless Communication in LEO Satellite Formation," *Proc. of the NASA/ESA Conf. on Adaptive Hardware and Systems*, 2008, pp. 255–262.

[177] M. Werner, et al., "Analysis of System Parameters for LEO/ICO-Satellite Communication Networks," *IEEE Journal of Selected Areas in Comm.*, vol. 12, no. 2, pp. 371–381, Feb. 1995.

[178] T. Peter and M. Ocef, "End-to-End Packet Delays in LEO Satellite Constellations," *Proc. of Int. Conf. on Systems, Signals, and Image Processing*, Jun. 25–28, 2008.

[179] T. Vladimirova, et al., "Development of a satellite sensor network for future space missions," *Proc. IEEE Aerospace Conf.*, Mar. 2008, Big Sky, MT.

[180] Y. Yang, et al., "Delay Distributions of Slotted ALOHA and CSMA," *IEEE Trans. on Comms*, vol. 51, no. 11, pp. 1846–1857, Nov. 2003.

[181] G. Bianchi, "Performance analysis of the IEEE 802.11 distributed coordination function," *IEEE Journal on Selected Areas in Comms*, vol. 18, no. 3, pp. 535–547, Mar. 2000.

[182] T. Sakurai and H. L. Vu, "MAC access delay of IEEE 802.11 DCF," *IEEE Trans. on Wireless Comms.*, Vol. 6, No. 5, pp. 1702–1710, May 2007.

[183] A. Ephremides and O. A. Mowafi, "Analysis of a Hybrid Access Scheme for Buffered Users-Probabilistic Time Division," *Trans. on Software Engineering*, vol. SE-8, no. 1, pp. 52–61, Jan. 1982.

[184] T. O. Walker, M. Tummala, and J. B. Michael, "A distributed medium access control protocol for wireless networks of cooperative radar systems," *Proc. 2007 IEEE Int. Conf. on System of Systems Engineering*, San Antonio, TX, Apr. 2007, pp. 1–6.

[185] T. O. Walker, M. Tummala, and J. McEachen, "Traffic-adaptive, flow-specific medium access for wireleess networks," Naval Postgraduate School, Monterey, CA, Tech. Rep. NPS-EC-09-002, Sep. 2009.

[186] T. O. Walker, M. Tummala, and J. McEachen, "Energy Efficiency of Centralized and Distributed Computation in Unattended Multi-hop Wireless Sensor Networks

for Battlefield Monitoring," *Proc. of the 43rd Annual Hawaii Int. Conf. on System Sciences*, Hawaii, Jan. 2010, accepted for publication.

[187] T. O. Walker, M. Tummala, and J. McEachen, " Energy-efficient, Flow-specific Medium Access using Preamble Sampling," *3rd International Conference on Signal Processing and Communication Systems*, Omaha, Nebraska, Sep. 28–30, 2009, accepted for publication.

[188] T. O. Walker, M. Tummala, J. McEachen and J. B. Michael, "Medium Access for Large-scale Wireless Networks for Missile Defense," *Proc. 2009 IEEE Int. Conf. on System of Systems Engineering*, Albuquerque, NM, Jun. 2009, pp. 1–5.

[189] T. O. Walker, M. Tummala, J. McEachen and J. B. Michael, "Flow-specific Medium Access for Networked Satellite Systems," *IEEE Systems Journal*, submitted for publication.

THIS PAGE INTENTIONALLY LEFT BLANK

APPENDIX. OPNET® SIMULATION CODE FOR TRAFFIC-ADAPTIVE CWS-MAC TRANSMITTER

In this appendix, we provide the significant portions of the OPNET® code used to generate the simulation results in this dissertation. Specifically, we provide the code used to model the traffic-adaptive CWS-MAC transmitter process and the switching process that transitions the data flow from one mode to the other. We begin with an overview schematic of the traffic-adaptive CWS-MAC transmitter node to understand where these processes reside.

A. TRAFFIC-ADAPTIVE CWS-MAC TRANSMITTER NODE SCHEMATIC

In Figure 96, we provide the schematic for the traffic-adaptive CWS-MAC transmitter node. Of particular interest are the transmitter process and the switching process, which reside in the *tx_proc* module and the *TA_switch_data* module (both seen in the figure), respectively. We provide the schematic and OPNET® simulation code for these two processes in the remaining sections of this appendix.

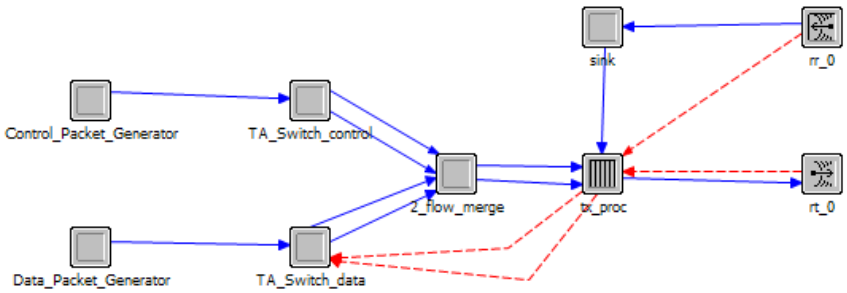


Figure 96. Traffic-adaptive CWS-MAC transmitter node model (TOW_ta_cws_node_tx_ack_thresh_wireless).

B. TRAFFIC-ADAPTIVE CWS-MAC TRANSMITTER PROCESS

1. Process Schematic

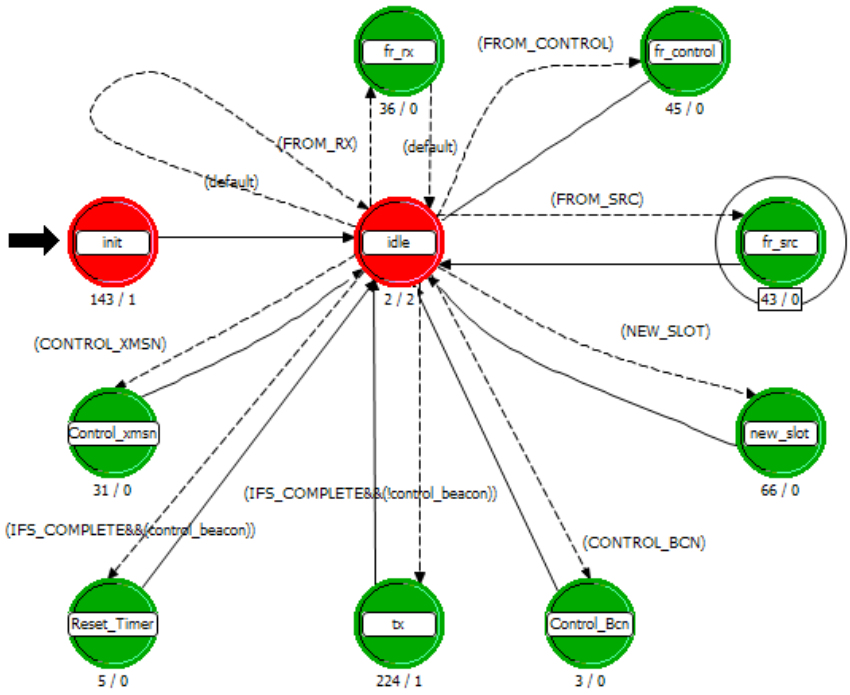


Figure 97. Traffic-adaptive CWS-MAC transmitter process model (TOW_ta_cws_slottedAloha_tx_ack_thresh).

2. Process Simulation Code

```
################################################################################
        Process Model Report:  TOW_ta_cws_slottedAloha_tx_ack_thresh
################################################################################

================================================================================
                          Process  Model Comments

Traffic-adaptive CWS-MAC transmitter process. Built from TDMA model.

================================================================================


================================================================================
                          Process Model Attributes
================================================================================
--------------------------------------------------------------------------------
                       Attribute: Switchover Threshold
--------------------------------------------------------------------------------
Data Type:          double
```

```
================================================================================
                        Process Model Interface Attributes
================================================================================
--------------------------------------------------------------------------------
                        Interface Attribute: begsim intrpt
--------------------------------------------------------------------------------
Assign Status:      hidden
Initial Value       enabled
Data Type:          toggle
Comments:           YES
          This attribute specifies whether a 'begin simulation interrupt' is generated for a processor  module's root
process at the start of the simulation.

--------------------------------------------------------------------------------
                          Interface Attribute: doc file
--------------------------------------------------------------------------------
Assign Status:      set
Initial Value       nd_module
Data Type:          string
Comments:           YES
          This attribute defines the name of the product help file which will be displayed when the user invokes help
for this object.

--------------------------------------------------------------------------------
                        Interface Attribute: endsim intrpt
--------------------------------------------------------------------------------
Assign Status:      hidden
Initial Value       disabled
Data Type:          toggle
Comments:           YES
          This attribute specifies whether an 'end simulation interrupt' is generated for a processor  module's root
process at the end of the simulation.

--------------------------------------------------------------------------------
                       Interface Attribute: failure intrpts
--------------------------------------------------------------------------------
Assign Status:      hidden
Initial Value       disabled
Data Type:          enumerated
Comments:           YES
          This attribute specifies whether failure interrupts are generated for a processor module's root process upon
failure of nodes or links in the network model.

--------------------------------------------------------------------------------
                       Interface Attribute: intrpt interval
--------------------------------------------------------------------------------
Assign Status:      hidden
Initial Value       disabled
Data Type:          toggle double
Comments:           YES
          This attribute specifies how often regular interrupts are scheduled for the root process of a processor
module.

--------------------------------------------------------------------------------
                          Interface Attribute: priority
--------------------------------------------------------------------------------
Assign Status:      hidden
Initial Value       0
Data Type:          integer
Comments:           YES
          This attribute is used to determine the execution order of events that are scheduled to occur at the same
simulation time.

--------------------------------------------------------------------------------
                      Interface Attribute: recovery intrpts
--------------------------------------------------------------------------------
Assign Status:      hidden
Initial Value       disabled
Data Type:          enumerated
Comments:           YES
          This attribute specifies whether recovery interrupts are scheduled for the processor module's root process
upon recovery of nodes or links in the network model.

--------------------------------------------------------------------------------
                          Interface Attribute: subqueue
--------------------------------------------------------------------------------
Assign Status:      hidden
Initial Value       (...)
Data Type:          compound
Comments:           YES
          This operation attribute permits the addition and deletion of subqueues within the queue module.

--------------------------------------------------------------------------------
                       Interface Attribute: super priority
--------------------------------------------------------------------------------
Assign Status:      hidden
Initial Value       disabled
Data Type:          toggle
Comments:           YES
          This attribute is used to determine the execution order of events that are scheduled to occur at the same
simulation time.
```

```
================================================================================
                        Process Model Global Attributes
================================================================================
--------------------------------------------------------------------------------
                             Attribute: Slot Length
--------------------------------------------------------------------------------
Data Type:          double

--------------------------------------------------------------------------------
                           Attribute: Transmission Rate
--------------------------------------------------------------------------------
Data Type:          double

--------------------------------------------------------------------------------
                              Attribute: Data IFS
--------------------------------------------------------------------------------
Data Type:          double

--------------------------------------------------------------------------------
                         Attribute: Control Minislot Length
--------------------------------------------------------------------------------
Data Type:          double

--------------------------------------------------------------------------------
                  Attribute: Number of Control Minislots per Slot
--------------------------------------------------------------------------------
Data Type:          integer

--------------------------------------------------------------------------------
                         Attribute: Control Beacon Duration
--------------------------------------------------------------------------------
Data Type:          double

--------------------------------------------------------------------------------
                         Attribute: Maximum Propagation Distance
--------------------------------------------------------------------------------
Data Type:          double

--------------------------------------------------------------------------------
                   Attribute: Slotted ALOHA Transmit Probability
--------------------------------------------------------------------------------
Data Type:          double

--------------------------------------------------------------------------------
                              Attribute: Upper Threshold
--------------------------------------------------------------------------------
Data Type:          double

--------------------------------------------------------------------------------
                              Attribute: Lower Threshold
--------------------------------------------------------------------------------
Data Type:          double

================================================================================
                                   Header Block
================================================================================
#include <math.h>

/* Constant Definitions */
#define ARQ_IN_STRM                     0
#define CONT_IN_STRM                    1
#define RX_IN_STRM                      2

#define TX_OUT_STRM                     0
#define ARQ_OUT_STRM                    1

#define TX_BUSY_STAT                    1
#define CH_BUSY_STAT                    0

#define SLOT_CODE                       5
#define IFS_CODE                        6
#define PULSE_CODE                      0
#define CONTROL_XMSN_CODE               7
#define CONTROL_BCN_CODE                8

#define CONTROL_SLOT                    1

#define DATA_PKT                        0
#define CONTROL_PKT                     1
#define ACK_PKT                         2

#define   PKT_TYPE_FIELD                0
#define   PKT_ID_FIELD                  1
#define   PKT_CREATION_TIME_FIELD       2
#define   NODE_ID_FIELD                 3
#define   PKT_XMSN_MODE                 10

#define   CONTENTION_MODE               1

#define DATA_QUEUE                      0
#define CONTROL_QUEUE                   1

#define EPSILON                         1e-10               /* rounding error factor */
#define TDMA_COMPLETE                   (-10)


/* Transition Condition Macros */
#define FROM_RX              (current_intrpt_type == OPC_INTRPT_STRM) && (op_intrpt_strm () == RX_IN_STRM)
#define FROM_SRC             (current_intrpt_type == OPC_INTRPT_STRM) && (op_intrpt_strm () == ARQ_IN_STRM)
#define FROM_CONTROL         (current_intrpt_type == OPC_INTRPT_STRM) && (op_intrpt_strm () == CONT_IN_STRM)

#define TRANSMITTING         (op_stat_local_read (TX_BUSY_STAT) == 1.0)
#define FREE                 (op_stat_local_read (CH_BUSY_STAT) == 0.0)

#define SLOT                 ((current_intrpt_type == OPC_INTRPT_SELF)&&(op_intrpt_code()==0))
```

```
#define NEW_SLOT              ((current_intrpt_type == OPC_INTRPT_SELF)&&(op_intrpt_code() == SLOT_CODE))
#define IFS_COMPLETE          ((current_intrpt_type == OPC_INTRPT_SELF)&&(op_intrpt_code() == IFS_CODE))
#define CONTROL_XMSN          ((current_intrpt_type == OPC_INTRPT_SELF)&&(op_intrpt_code() == CONTROL_XMSN_CODE))
#define CONTROL_BCN           ((current_intrpt_type == OPC_INTRPT_SELF)&&(op_intrpt_code() == CONTROL_BCN_CODE))
#define MY_SLOT               (op_sim_time() < my_slot_expiration_time)

#define END                   (current_intrpt_type == OPC_INTRPT_STAT)

#define DATA_ENQ              (!(op_subq_empty (DATA_QUEUE)))
#define CONTROL_DATA_ENQ      (!(op_subq_empty (CONTROL_QUEUE)))

#define CHANGERATE            ((current_intrpt_type == OPC_INTRPT_SELF)&&(op_intrpt_code()==1))
#define   SELF_INTRPT_SCHLD   (intrpt_flag == 1)
#define NOT_FIRST             (op_pk_get (ARQ_IN_STRM)>160)

/* Global Variables */
int       tdma_pk_sent;
int       tdma_pk_rcvd;
int       tdma_bits_sent;
int       tdma_bits_rcvd;
int       tdma_setup;
int       tdma_id;
int       num_slots;
int       control_beacon;     /* control beacon flag used to seize slot as control slot */

================================================================================
                              State Variable Block
================================================================================
int       \my_offset;

double    \slot_length;

int       \intrpt_flag;

int       \num_pk_sent;

int       \num_pk_rcvd;

int       \num_bits_sent;

int       \num_bits_rcvd;

Stathandle          \num_pk_sent_stat;

Stathandle          \global_pk_sent_stat;

Stathandle          \num_pk_rcvd_stat;

Stathandle          \global_pk_rcvd_stat;

Objid     \my_node_id;

Objid     \my_id;

Stathandle          \num_bits_sent_stat;

Stathandle          \global_bits_sent_stat;

Stathandle          \num_bits_rcvd_stat;

Stathandle          \global_bits_rcvd_stat;

Stathandle          \bits_sec_rcvd_stat;

Stathandle          \bits_sec_sent_stat;

Stathandle          \pk_sec_rcvd_stat;

Stathandle          \pk_sec_sent_stat;

Stathandle          \global_bits_sec_rcvd_stat;

Stathandle          \global_bits_sec_sent_stat;

Stathandle          \global_pk_sec_rcvd_stat;

Stathandle          \global_pk_sec_sent_stat;

Stathandle          \current_offset_stat;

Stathandle          \transmission_rate_stat;

int       \count;

int       \ack_to_sendg;

int       \rn_to_sendg;

int       \tsn;

int       \trn;

int       \changer;

Objid     \tx_ch_id;

double    \actual_time;

int       \percent_lost;

int       \pk_counter;

double    \data_ifs;

double    \tx_data_rate;
```

```
double    \my_slot_expiration_time;

/* Number of control minislots in a slot */
int       \num_minislots;

/* Length of control minislot (sec) */
double    \control_minislot_length;

/* Duration of Control beacon transmission (sec) */
double    \control_beacon_length;

/* Boolean flag to indicate switchover has occurred */
int       \switch_flag;

/* Time at which switchover occurs */
double    \threshold;

/* Guard band to allow for tprop in slot tranmsissions, based on max prop distance. */
double    \guard_time;

/* Maximum propagation distance for setting guard band */
double    \max_prop_dist;

/* Current control minislot */
int       \control_minislot;

/* Slotted ALOHA transmission probability */
double    \xmt_prob;

/* Flag to indicate whether or not we are waiting for an acknowledgement (1 = waiting). */
int       \awaiting_ACK;

/* Contention queue state.  0 = lower state (below thresholds).  1 = upper state (above thresholds). */
Stathandle          \contention_queue_state;

/* Non-contention queue state.  0 = lower state (below threshold).  1 = upper state (above threshold). */
Stathandle          \noncontention_queue_state;

/* Upper queue threshold for mode switchover. */
double    \upper_threshold;

/* Lower queue threshold for mode switchover. */
double    \lower_threshold;

/* Flag to indicate whether or not queue should be flushed.  0 = flush, 1 = already flushed - NO! */
int       \flush_flag;

================================================================================
                            Temporary Variable Block
================================================================================
Packet*   pkptr;
Packet*   cp_pkptr;
Packet* pkptr1;

Objid     current_node_id, tx_id, comp_id;

double    floor();
double    fmod();

int       used_slots;
int       current_offset;
int       next_offset;
int       i;


double    current_time;
double    time_left_in_slot;
double    pk_len;
double    pk_time;

double    my_next_slot_time;

int       current_intrpt_type;

int       num_fixed, num_mobile, num_sat;

Objid     ack_node_id;
Objid     pkt_id;
int       pkt_type;

int       subqueuesize;                         /* Temp variable that holds subqueue size */


================================================================================
                                Diagnostic Block
================================================================================
printf ("Object ID = %d Current Sim Time = %g\n", my_id, op_sim_time ());
printf ("My TDMA Offset = %d\n", my_offset);
printf ("Number of TDMA Slots = %d\n", num_slots);
printf ("Number of Packets Received = %d\n", num_pk_rcvd);
printf ("Number of Bits Received = %d\n", num_bits_rcvd);
printf ("Number of Packets Sent = %d\n", num_pk_sent);
printf ("Number of Bits Sent = %d\n", num_bits_sent);

================================================================================
                  Enter Execs for the unforced state "init"
================================================================================
/* Get inital info for all nodes */
op_ima_sim_attr_get (OPC_IMA_DOUBLE, "Slot Length", &slot_length);
op_ima_sim_attr_get (OPC_IMA_DOUBLE, "Control Minislot Length", &control_minislot_length);
op_ima_sim_attr_get (OPC_IMA_INTEGER, "Number of Control Minislots per Slot", &num_minislots);
op_ima_sim_attr_get (OPC_IMA_DOUBLE, "Control Beacon Duration", &control_beacon_length);
op_ima_sim_attr_get (OPC_IMA_DOUBLE, "Data IFS", &data_ifs);
```

```
op_ima_sim_attr_get (OPC_IMA_DOUBLE, "Transmission Rate", &tx_data_rate);
op_ima_sim_attr_get (OPC_IMA_DOUBLE, "Switchover Threshold", &threshold);
op_ima_sim_attr_get (OPC_IMA_DOUBLE, "Maximum Propagation Distance", &max_prop_dist);
op_ima_sim_attr_get (OPC_IMA_DOUBLE, "Upper Threshold", &upper_threshold);
op_ima_sim_attr_get (OPC_IMA_DOUBLE, "Lower Threshold", &lower_threshold);
op_ima_sim_attr_get (OPC_IMA_DOUBLE, "Slotted ALOHA Transmit Probability", &xmt_prob);

/* Test attributes */
printf("Upper Threshold is set at = %f \n", upper_threshold);
printf("Slot Length is set at = %f \n", slot_length);
printf("Control Minislot Length is set at = %f \n", control_minislot_length);
printf("Number of Control Minislots per Slot is set at = %i \n", num_minislots);
printf("Control Beacon Duration is set at = %f \n", control_beacon_length);
printf("Data IFS is set at = %f \n", data_ifs);
printf("Transmission Rate is set at = %f \n", tx_data_rate);
printf("Switchover Threshold is set at = %i \n", threshold);
printf("Maximum Propagation Distance is set at = %f \n", max_prop_dist);
printf("Lower Threshold is set at = %f \n", lower_threshold);
printf("Slotted ALOHA Transmit Probability is set at = %f \n", xmt_prob);


/* Set guard time */
guard_time = (max_prop_dist/3e8)+0.1*(max_prop_dist/3e8);
printf("Guard time = %f \n", guard_time);

/* Set switchover time and flag and awaiting_ACK flag */
switch_flag = 0;
awaiting_ACK = 0;
flush_flag = 0;


/* Classic TDMA */
//data_ifs = 0;
//num_minislots = 0;

tsn=0;
trn=0;
percent_lost = 0;
transmission_rate_stat = op_stat_reg("Transmission rate", OPC_STAT_INDEX_NONE,OPC_STAT_LOCAL);
op_stat_write (transmission_rate_stat,tx_data_rate);
changer = 1;

my_id = op_id_self();
my_node_id = op_topo_parent (my_id);

actual_time=0;

/* Initialize tdma offsets if not done previously */
if (tdma_setup != TDMA_COMPLETE)
{
          num_slots = 0;
          tdma_setup = TDMA_COMPLETE;
}

/* Calculate the offset for this node */
num_slots = num_slots+1;
my_offset = num_slots-1;


/* Set interrupt for arrival of next slot */
op_intrpt_schedule_self (op_sim_time () + slot_length, SLOT_CODE);


if (op_prg_odb_ltrace_active ("tdma"))
{
          printf ("Node Objid = %d\n", my_node_id);
          printf ("Node Offset = %d\n", my_offset);
          printf ("\n");
}


/* Initialize statistic calculation variables */

/*
tdma_pk_sent = 0;
tdma_pk_rcvd = 0;
tdma_bits_sent = 0;
tdma_bits_rcvd = 0;
num_pk_sent = 0;
num_pk_rcvd = 0;
num_bits_sent = 0;
num_bits_rcvd = 0;
*/

/* Register Statistics */
contention_queue_state = op_stat_reg ("Contention Queue State", OPC_STAT_INDEX_NONE, OPC_STAT_LOCAL);
noncontention_queue_state = op_stat_reg ("Non-contention Queue State", OPC_STAT_INDEX_NONE, OPC_STAT_LOCAL);

num_pk_sent_stat = op_stat_reg ("TDMA.Load (packets)", OPC_STAT_INDEX_NONE, OPC_STAT_LOCAL);
global_pk_sent_stat = op_stat_reg ("TDMA.TDMA Load (packets)", OPC_STAT_INDEX_NONE, OPC_STAT_GLOBAL);
num_pk_rcvd_stat = op_stat_reg ("TDMA.Traffic Received (packets)", OPC_STAT_INDEX_NONE, OPC_STAT_LOCAL);
global_pk_rcvd_stat = op_stat_reg ("TDMA.TDMA Traffic Received (packets)", OPC_STAT_INDEX_NONE, OPC_STAT_GLOBAL);

num_bits_sent_stat = op_stat_reg ("TDMA.Load (bits)", OPC_STAT_INDEX_NONE, OPC_STAT_LOCAL);
global_bits_sent_stat = op_stat_reg ("TDMA.TDMA Load (bits)", OPC_STAT_INDEX_NONE, OPC_STAT_GLOBAL);
num_bits_rcvd_stat = op_stat_reg ("TDMA.Traffic Received (bits)", OPC_STAT_INDEX_NONE, OPC_STAT_LOCAL);
global_bits_rcvd_stat = op_stat_reg ("TDMA.TDMA Traffic Received (bits)", OPC_STAT_INDEX_NONE, OPC_STAT_GLOBAL);

bits_sec_rcvd_stat = op_stat_reg ("TDMA.Traffic Received (bits/sec)", OPC_STAT_INDEX_NONE, OPC_STAT_LOCAL);
bits_sec_sent_stat = op_stat_reg ("TDMA.Load (bits/sec)", OPC_STAT_INDEX_NONE, OPC_STAT_LOCAL);
pk_sec_rcvd_stat = op_stat_reg ("TDMA.Traffic Received (packets/sec)", OPC_STAT_INDEX_NONE, OPC_STAT_LOCAL);
pk_sec_sent_stat = op_stat_reg ("TDMA.Load (packets/sec)", OPC_STAT_INDEX_NONE, OPC_STAT_LOCAL);

global_bits_sec_rcvd_stat = op_stat_reg ("TDMA.TDMA Traffic Received (bits/sec)", OPC_STAT_INDEX_NONE,
OPC_STAT_GLOBAL);
```

```
global_pk_sec_rcvd_stat = op_stat_reg ("TDMA.TDMA Traffic Received (bits/sec)", OPC_STAT_INDEX_NONE, OPC_STAT_GLOBAL);
global_pk_sec_sent_stat = op_stat_reg ("TDMA.TDMA Load (packets/sec)", OPC_STAT_INDEX_NONE, OPC_STAT_GLOBAL);
global_bits_sec_sent_stat = op_stat_reg ("TDMA.TDMA Load (packets/sec)", OPC_STAT_INDEX_NONE, OPC_STAT_GLOBAL);


current_offset_stat = op_stat_reg("current_offset", OPC_STAT_INDEX_NONE,OPC_STAT_LOCAL);


/* Initialize queue state statistics */
op_stat_write(contention_queue_state, 0.0);
op_stat_write(noncontention_queue_state, 0.0);


/* Schedule interupt to complete initialization in the exit execs */
op_intrpt_schedule_self (op_sim_time (), 0);

--------------------------------------------------------------------------------
================================================================================
                    Exit Execs for the unforced state "init"
================================================================================

--------------------------------------------------------------------------------
================================================================================
                            transition   init -> idle
================================================================================
name:     tr_12
condition:
executive:
color:    black
drawing style:      spline
doc file: pr_transition

--------------------------------------------------------------------------------


================================================================================
                   Enter Execs for the unforced state "idle"
================================================================================
--------------------------------------------------------------------------------
================================================================================
                    Exit Execs for the unforced state "idle"
================================================================================
current_intrpt_type = op_intrpt_type();

--------------------------------------------------------------------------------
================================================================================
                            transition   idle -> fr_rx
================================================================================
name:     tr_13
condition:          FROM_RX
executive:
color:    black
drawing style:      spline
doc file: pr_transition

--------------------------------------------------------------------------------

================================================================================
                           transition   idle -> fr_src
================================================================================
name:     tr_15
condition:          FROM_SRC
executive:
color:    black
drawing style:      spline
doc file: pr_transition

--------------------------------------------------------------------------------

================================================================================
                             transition   idle -> tx
================================================================================
name:     tr_19
condition:          IFS_COMPLETE&&(!control_beacon)
executive:
color:    black
drawing style:      spline
doc file: pr_transition

--------------------------------------------------------------------------------

================================================================================
                            transition   idle -> idle
================================================================================
name:     tr_58_0
condition:          default
executive:
color:    black
drawing style:      spline
doc file: pr_transition

--------------------------------------------------------------------------------

================================================================================
                          transition   idle -> new_slot
================================================================================
name:     tr_60
condition:          NEW_SLOT
executive:
color:    black
drawing style:      spline
doc file: pr_transition

--------------------------------------------------------------------------------
```

```
================================================================================
                        transition   idle -> fr_control
================================================================================
name:     tr_64
condition:          FROM_CONTROL
executive:
color:    black
drawing style:      spline
doc file: pr_transition

--------------------------------------------------------------------------------

================================================================================
                       transition   idle -> Control_xmsn
================================================================================
name:     tr_66
condition:          CONTROL_XMSN
executive:
color:    black
drawing style:      spline
doc file: pr_transition

--------------------------------------------------------------------------------

================================================================================
                        transition   idle -> Control_Bcn
================================================================================
name:     tr_71
condition:          CONTROL_BCN
executive:
color:    black
drawing style:      spline
doc file: pr_transition

--------------------------------------------------------------------------------

================================================================================
                        transition   idle -> Reset_Timer
================================================================================
name:     tr_73
condition:          IFS_COMPLETE&&(control_beacon)
executive:
color:    black
drawing style:      spline
doc file: pr_transition

--------------------------------------------------------------------------------

================================================================================
                    Enter Execs for the forced state "fr_rx"
================================================================================
/* Obtain the incoming packet.          */
pkptr =  op_pk_get(RX_IN_STRM);

/* Get packet info */
op_pk_fd_get (pkptr, PKT_TYPE_FIELD, &pkt_type);
op_pk_fd_get (pkptr, NODE_ID_FIELD, &ack_node_id);
op_pk_fd_get (pkptr, PKT_ID_FIELD, &pkt_id);

if ((pkt_type == ACK_PKT) && (ack_node_id == my_node_id))
          {
          printf("** TIME: %f, ACK PACKET %d RECEIVED AT TX PROCESS AT NODE %d\n",op_sim_time(),pkt_id,my_node_id);

          /* Pull packet form head of contention queue and delete it and reset awaiting_ACK flag */
          cp_pkptr = op_subq_pk_remove (CONTROL_QUEUE, OPC_QPOS_HEAD);
          op_pk_destroy(cp_pkptr);
          awaiting_ACK = 0;

          /* Check contention queue status, set state to 0.0 if below lower threshold */
          subqueuesize = op_subq_stat (CONTROL_QUEUE, OPC_QSTAT_PKSIZE);
          if (subqueuesize < lower_threshold)
                    {
                    op_stat_write(contention_queue_state,0.0);
                    }
          }

op_pk_destroy(pkptr);


--------------------------------------------------------------------------------
================================================================================
                    Exit Execs for the forced state "fr_rx"
================================================================================
NONE
--------------------------------------------------------------------------------
================================================================================
                           transition   fr_rx -> idle
================================================================================
name:     tr_14
condition:          default
executive:
color:    black
drawing style:      spline
doc file: pr_transition

--------------------------------------------------------------------------------

================================================================================
                   Enter Execs for the forced state "fr_src"
================================================================================
int tmp_sn;

pkptr =  op_pk_get (ARQ_IN_STRM);

/* Queue Control Packet */
```

```
op_subq_pk_insert (DATA_QUEUE, pkptr, OPC_QPOS_TAIL);
printf("** TIME: %f, DATA PACKET %d",op_sim_time(),op_pk_id(pkptr));
printf(" QUEUED AT NODE %d\n",my_node_id);

/* Check non-contention queue status, set state to 1.0 if above upper threshold */
subqueuesize = op_subq_stat (CONTROL_QUEUE, OPC_QSTAT_PKSIZE);
if (subqueuesize > upper_threshold)
          {
          op_stat_write(noncontention_queue_state,1.0);
          }
--------------------------------------------------------------------------------

================================================================================
                    Exit Execs for the forced state "fr_src"
================================================================================
NONE
--------------------------------------------------------------------------------
================================================================================
                             transition   fr_src -> idle
================================================================================
name:     tr_54
condition:
executive:
color:    black
drawing style:      spline
doc file: pr_transition

--------------------------------------------------------------------------------

================================================================================
                     Enter Execs for the forced state "tx"
================================================================================
int tmp_sn;
int i;

current_time = op_sim_time();

/* See if there is time left in the current slot to transmit */
/* the packet at the top of the queue */

time_left_in_slot = (my_slot_expiration_time - guard_time) - current_time;
//printf("TIME LEFT IN SLOT AT NODE %d : %f\n",my_node_id,time_left_in_slot);

pk_len = (double) op_pk_total_size_get (op_subq_pk_access (0, OPC_QPOS_HEAD));
pk_time = (double) pk_len / tx_data_rate;


printf("TIME: %f;  CHANNEL READY --- NODE: %d --- slot expiration time::%f\n",
current_time,my_node_id,my_slot_expiration_time);
printf("           pk_time:%f --- time_left:%f\n", pk_time,time_left_in_slot);

/* If this is my slot and I have enough time to transmit the */
/* entire packet then transmit. Otherwise set a self intrpt  */
/* for the beginning of my next slot.                        */
while((op_sim_time() < my_slot_expiration_time) && (pk_time < time_left_in_slot) && (DATA_ENQ) && (!control_beacon))
          {

          /* dequeue the packet and send it */
          pkptr = op_subq_pk_remove (DATA_QUEUE, OPC_QPOS_HEAD);

          printf("  ~~  DATA PACKET %d REMOVED FROM QUEUE for TX\n",op_pk_id(pkptr));

          /* Check non-contention queue status, set state to 0.0 if below lower threshold */
          subqueuesize = op_subq_stat (DATA_QUEUE, OPC_QSTAT_PKSIZE);
          if (subqueuesize < lower_threshold)
                    {
                    op_stat_write(noncontention_queue_state,0.0);
                    }

          /* reset the flag to schedule a self interrupt */
          /* for packets arriving subsequent to this one */
          intrpt_flag = 0;

          pk_len = (double) op_pk_total_size_get (pkptr);
          time_left_in_slot = (time_left_in_slot - pk_time);

          /** Record Statistics **/
          /** The bits/sec or packets/sec statistics are recorded in           **/
          /** bits and packets, and then the OPNET statistic "capture          **/
          /** mode" is used to obtain a bucketized sum over time.                      **/
          /** Record extra 0.0 data-points to enable proper computation        **/
          /** of the "sum/time" based statistics.                                                  **/

          op_stat_write (num_pk_sent_stat, 1.0);
          op_stat_write (pk_sec_sent_stat, 1.0);
          op_stat_write (pk_sec_sent_stat, 0.0);

          op_stat_write (global_pk_sent_stat, 1.0);
          op_stat_write (global_pk_sec_sent_stat, 1.0);
          op_stat_write (global_pk_sec_sent_stat, 0.0);
          op_stat_write (num_bits_sent_stat, pk_len);
          op_stat_write (bits_sec_sent_stat, pk_len);
          op_stat_write (bits_sec_sent_stat, 0.0);

          op_stat_write (global_bits_sent_stat, pk_len);
          op_stat_write (global_bits_sec_sent_stat, pk_len);
          op_stat_write (global_bits_sec_sent_stat, 0.0);


          cp_pkptr = op_pk_copy(pkptr);
          op_pk_fd_get(cp_pkptr,2,&tsn);
          op_pk_fd_get(cp_pkptr,3,&trn);

          op_pk_destroy(cp_pkptr);
```

```
                    if (pk_counter < percent_lost)
                    {
                              op_pk_destroy(pkptr);
                              //printf("time %f: LOST packet %d at node %d\n", op_sim_time(),tsn,my_node_id);

                              }
                    else
                              {
                              op_pk_send (pkptr, TX_OUT_STRM);
                              printf("  ~~  DATA PACKET %d",op_pk_id(pkptr));
                              printf(" TRANSMITTED from NODE %d\n",my_node_id);
                              }

           pk_counter = ((pk_counter+1)%100);

          } /* End while */


          next_offset = my_offset - current_offset;

          if (next_offset <= 0)
                    {
                    next_offset += num_slots;
                    }


          my_next_slot_time = (double) (used_slots + next_offset) * slot_length;


          {
          /* schedule a self interrupt only if one has not */
          /* been already scheduled for the next slot time */
          if ((!SELF_INTRPT_SCHLD)&&((current_offset != my_offset) || (pk_time > time_left_in_slot))&&(DATA_ENQ))
                    {
                    intrpt_flag = 1;
                    }}
--------------------------------------------------------------------------------
================================================================================
                      Exit Execs for the forced state "tx"
================================================================================

--------------------------------------------------------------------------------
================================================================================
                             transition   tx -> idle
================================================================================
name:     tr_18
condition:
executive:
color:    black
drawing style:      spline
doc file: pr_transition

--------------------------------------------------------------------------------


================================================================================
                  Enter Execs for the forced state "new_slot"
================================================================================
current_time = op_sim_time();


//printf("TIME: %f, NODE %d REPORTS ARRIVAL OF NEW SLOT\n",op_sim_time(),my_node_id);


/* Reset interrupt for arrival of next slot */
op_intrpt_schedule_self (op_sim_time () + slot_length, SLOT_CODE);

/* Check if control packets are queued */
/* If so, sieze slot as control frame and set interrupt to clear beacon  */
if (CONTROL_DATA_ENQ)
          {
          if (control_beacon != CONTROL_SLOT)
                    {
                    control_beacon = CONTROL_SLOT;
                    op_intrpt_schedule_self (op_sim_time () + control_beacon_length, CONTROL_BCN_CODE);
                    }

          control_minislot = 0;

          /* Set interupt for first control minislot */
          op_intrpt_schedule_self ((op_sim_time () + control_beacon_length), CONTROL_XMSN_CODE);

          }


/* Determine if current slot is my slot */
used_slots = (int) floor ((current_time / slot_length) + EPSILON);
current_offset = used_slots % num_slots;
if (current_offset == my_offset){
          my_slot_expiration_time = current_time + slot_length;
          printf("TIME: %f, NODE %d reports this is MY SLOT",op_sim_time(),my_node_id);
          printf(" with SLOT EXPIRATION TIME: %f\n",my_slot_expiration_time);
          }

/* If so and data packets queued, */
/* Set interrupt for IFS */
if (DATA_ENQ && (current_offset == my_offset)){
          op_intrpt_schedule_self (op_sim_time () + data_ifs, IFS_CODE);
          printf("          IFS intrpt set for %f\n",op_sim_time()+data_ifs);
                    //printf("           Current offset: %d;", current_offset);
                    //printf("  My offset: %d;", my_offset);
                    //printf("  Used slots: %d\n", used_slots);

          }
```

```
--------------------------------------------------------------------------------
================================================================================
                  Exit Execs for the forced state "new_slot"
================================================================================
NONE
--------------------------------------------------------------------------------
================================================================================
                          transition   new_slot -> idle
================================================================================
name:     tr_61
condition:
executive:
color:    black
drawing style:      spline
doc file: pr_transition

--------------------------------------------------------------------------------


================================================================================
                Enter Execs for the forced state "fr_control"
================================================================================
pkptr =  op_pk_get (CONT_IN_STRM);

/* Queue Control Packet */

/* CWS-MAC */
op_subq_pk_insert (CONTROL_QUEUE, pkptr, OPC_QPOS_TAIL);

/* Check contention queue status, set state to 1.0 if above upper threshold */
subqueuesize = op_subq_stat (CONTROL_QUEUE, OPC_QSTAT_PKSIZE);
if (subqueuesize > upper_threshold)
          {
          op_stat_write(contention_queue_state,1.0);
          if (flush_flag == 0)
                    {
                    /*  flush queue of data packets */
                    for (i = subqueuesize; i > 1; i--)
                              {
                              /* Check for data packet */
                              pkptr1 = op_subq_pk_access (CONTROL_QUEUE, i-1);
                              op_pk_fd_get (pkptr1, PKT_TYPE_FIELD, &pkt_type);
                              if (pkt_type == DATA_PKT)
                                        {
                                        /* If data packet, move to data queue */
                                        pkptr1 = op_subq_pk_remove (CONTROL_QUEUE, i-1);
                                        op_subq_pk_insert (DATA_QUEUE, pkptr1, OPC_QPOS_HEAD);
                                        }
                              }
                    flush_flag = 1;
                    }
          }

/* classic TDMA */
//op_subq_pk_insert (DATA_QUEUE, pkptr, OPC_QPOS_HEAD);

printf("** TIME: %f, CONTROL PACKET %d QUEUED",op_sim_time(),op_pk_id (pkptr));
printf(" AT NODE %d\n",my_node_id);


--------------------------------------------------------------------------------
================================================================================
                 Exit Execs for the forced state "fr_control"
================================================================================
NONE
--------------------------------------------------------------------------------
================================================================================
                          transition   fr_control -> idle
================================================================================
name:     tr_65
condition:
executive:
color:    black
drawing style:      spline
doc file: pr_transition

--------------------------------------------------------------------------------

================================================================================
               Enter Execs for the forced state "Control_xmsn"
================================================================================
// pkptr = op_subq_pk_access (CONTROL_QUEUE, OPC_QPOS_HEAD);
// printf("** TIME: %f, CONTROL PACKET %d PENDING",op_sim_time(),op_pk_id(pkptr));
// printf(" AT NODE %d WITH ASSIGNED MINISLOT OF %d\n",my_node_id,control_minislot);

control_minislot++;

/* Use uniform distribution to contend for transmission */
if ((((op_dist_uniform (100))/100) < xmt_prob) && !op_subq_empty (CONTROL_QUEUE))
          {
          /* Medium access is successful, copy contention packet and tranmsit it */
          pkptr = op_subq_pk_access (CONTROL_QUEUE, OPC_QPOS_HEAD);
          cp_pkptr = op_pk_copy(pkptr);

          /* Get packet ID and label as contention mode packet */
          op_pk_fd_get (cp_pkptr, PKT_ID_FIELD, &pkt_id);
          op_pk_fd_set (cp_pkptr, PKT_XMSN_MODE, OPC_FIELD_TYPE_INTEGER, CONTENTION_MODE, 0);

          /* Set flag to indicate awaiting ACK */
          awaiting_ACK = 1;

          printf("** TIME: %f, CONTENTION MODE PACKET %d TRANSMITTED",op_sim_time(),pkt_id);
          printf(" FROM NODE %d\n",my_node_id);
          op_pk_send (cp_pkptr, TX_OUT_STRM);
          }
```

```
/* Reset interrupt for next control minislot */
if (!op_subq_empty (CONTROL_QUEUE) && (control_minislot != num_minislots))
          {
          op_intrpt_schedule_self ((op_sim_time () + control_minislot_length), CONTROL_XMSN_CODE);
          }

--------------------------------------------------------------------------------
================================================================================
                Exit Execs for the forced state "Control_xmsn"
================================================================================
NONE
--------------------------------------------------------------------------------
================================================================================
                        transition   Control_xmsn -> idle
================================================================================
name:     tr_68
condition:
executive:
color:    black
drawing style:      spline
doc file: pr_transition

--------------------------------------------------------------------------------


================================================================================
                Enter Execs for the forced state "Control_Bcn"
================================================================================
control_beacon = !CONTROL_SLOT;
printf("** TIME: %f, CONTROL BEACON TERMINATED",op_sim_time());
printf(" AT NODE %d\n",my_node_id);
--------------------------------------------------------------------------------
================================================================================
                Exit Execs for the forced state "Control_Bcn"
================================================================================
NONE
--------------------------------------------------------------------------------
================================================================================
                          transition   Control_Bcn -> idle
================================================================================
name:     tr_72
condition:
executive:
color:    black
drawing style:      spline
doc file: pr_transition

--------------------------------------------------------------------------------


================================================================================
                Enter Execs for the forced state "Reset_Timer"
================================================================================
/* Reset Timer to transmit data at end of Control Minislots */
op_intrpt_schedule_self (op_sim_time () + (num_minislots*control_minislot_length - data_ifs + EPSILON), IFS_CODE);
--------------------------------------------------------------------------------
================================================================================
                Exit Execs for the forced state "Reset_Timer"
================================================================================
NONE
--------------------------------------------------------------------------------
================================================================================
                          transition   Reset_Timer -> idle
================================================================================
name:     tr_74
condition:
executive:
color:    black
drawing style:      spline
doc file: pr_transition

--------------------------------------------------------------------------------
```

C. TRAFFIC-ADAPTIVE CWS-MAC SWITCH PROCESS

1. Process Schematic

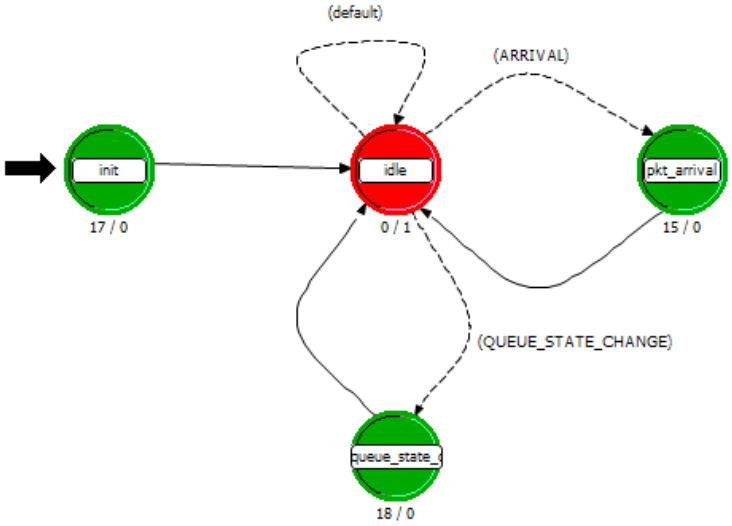


Figure 98. Traffic-adaptive CWS-MAC switching process (TOW_ta_switch_threshold_data).

2. Process Simulation Code

```
###############################################################################
                Process Model Report:  TOW_ta_switch_thresh_data
###############################################################################

===============================================================================
                             Process Model Attributes
===============================================================================
-------------------------------------------------------------------------------
                               Attribute: Threshold
-------------------------------------------------------------------------------
Data Type:          double
Comments:
          Switchover threshold for traffic adaptive mechanism (sec)

-------------------------------------------------------------------------------
                    Attribute: Fix Data in Non-contention Mode
-------------------------------------------------------------------------------
Data Type:          integer
Comments:
          Flag used to fix the data in the non-contention mode.  0 = allow data flow to move between modes. 1 = fix
data in non-contention mode.

===============================================================================
                         Process Model Interface Attributes
===============================================================================
-------------------------------------------------------------------------------
                         Interface Attribute: begsim intrpt
-------------------------------------------------------------------------------
Assign Status:      set
Initial Value       enabled
Data Type:          toggle
Comments:           YES
          This attribute specifies whether a 'begin simulation interrupt' is generated for a processor  module's root
process at the start of the simulation.

-------------------------------------------------------------------------------
                           Interface Attribute: doc file
-------------------------------------------------------------------------------
Assign Status:      set
Initial Value       nd_module
Data Type:          string
```

```
Comments:           YES
          This attribute defines the name of the product help file which will be displayed when the user invokes help
for this object.

--------------------------------------------------------------------------------
                        Interface Attribute: endsim intrpt
--------------------------------------------------------------------------------
Assign Status:      set
Initial Value       disabled
Data Type:          toggle
Comments:           YES
          This attribute specifies whether an 'end simulation interrupt' is generated for a processor  module's root
process at the end of the simulation.

--------------------------------------------------------------------------------
                       Interface Attribute: failure intrpts
--------------------------------------------------------------------------------
Assign Status:      set
Initial Value       disabled
Data Type:          enumerated
Comments:           YES
          This attribute specifies whether failure interrupts are generated for a processor module's root process upon
failure of nodes or links in the network model.

--------------------------------------------------------------------------------
                       Interface Attribute: intrpt interval
--------------------------------------------------------------------------------
Assign Status:      set
Initial Value       disabled
Data Type:          toggle double
Comments:           YES
          This attribute specifies how often regular interrupts are scheduled for the root process of a processor
module.

--------------------------------------------------------------------------------
                          Interface Attribute: priority
--------------------------------------------------------------------------------
Assign Status:      set
Initial Value       0
Data Type:          integer
Comments:           YES
          This attribute is used to determine the execution order of events that are scheduled to occur at the same
simulation time.

--------------------------------------------------------------------------------
                        Interface Attribute: recovery intrpts
--------------------------------------------------------------------------------
Assign Status:      set
Initial Value       disabled
Data Type:          enumerated
Comments:           YES
          This attribute specifies whether recovery interrupts are scheduled for the processor module's root process
upon recovery of nodes or links in the network model.

--------------------------------------------------------------------------------
                          Interface Attribute: subqueue
--------------------------------------------------------------------------------
Assign Status:      set
Initial Value       (...)
Data Type:          compound
Comments:           YES
          This operation attribute permits the addition and deletion of subqueues within the queue module.

--------------------------------------------------------------------------------
                       Interface Attribute: super priority
--------------------------------------------------------------------------------
Assign Status:      set
Initial Value       disabled
Data Type:          toggle
Comments:           YES
          This attribute is used to determine the execution order of events that are scheduled to occur at the same
simulation time.


================================================================================
                         Process Model Global Attributes
================================================================================
--------------------------------------------------------------------------------
                       Attribute: Maximum Propagation Distance
--------------------------------------------------------------------------------
Data Type:          double


================================================================================
                                    Header Block
================================================================================
/* Constant Definitions */
#define CONT_OUT_STRM                   0
#define NONCONT_OUT_STRM                1
#define CONT_STAT_STRM                  0
#define NONCONT_STAT_STRM               1


/* Transition Condition Macros */
#define ARRIVAL                         (current_intrpt_type == OPC_INTRPT_STRM)
#define QUEUE_STATE_CHANGE              (current_intrpt_type == OPC_INTRPT_STAT)
```

```
================================================================================
                              State Variable Block
================================================================================
/* Time at which access mode swithces over */
double    \threshold;

double    \max_prop_dist;

/* 0 = lower mode,  1 = upper mode. */
int       \upper_mode_flag;

/* 0=allow data to move between modes, 1=fix data flow in non-contention mode */
int       \fix_data_in_non_cont;

================================================================================
                            Temporary Variable Block
================================================================================
Packet*   pkptr;

int                current_intrpt_type;
int                current_intrpt_src;
================================================================================
                    Enter Execs for the forced state "init"
================================================================================
op_ima_sim_attr_get (OPC_IMA_DOUBLE, "Threshold", &threshold);
op_ima_sim_attr_get (OPC_IMA_INTEGER, "Fix Data in Non-contention Mode", &fix_data_in_non_cont);

printf("Fix Data in Non-contention Mode is set at = %i \n", fix_data_in_non_cont);


upper_mode_flag = 0;

if(fix_data_in_non_cont)
          {
          upper_mode_flag = 1;
          }

--------------------------------------------------------------------------------
================================================================================
                     Exit Execs for the forced state "init"
================================================================================
NONE
--------------------------------------------------------------------------------
================================================================================
                            transition   init -> idle
================================================================================
name:     tr_0
condition:
executive:
color:    black
drawing style:      spline
doc file: pr_transition

--------------------------------------------------------------------------------


================================================================================
                   Enter Execs for the unforced state "idle"
================================================================================
NONE
--------------------------------------------------------------------------------
================================================================================
                    Exit Execs for the unforced state "idle"
================================================================================
current_intrpt_type = op_intrpt_type();
--------------------------------------------------------------------------------
================================================================================
                        transition   idle -> pkt_arrival
================================================================================
name:     tr_1
condition:          ARRIVAL
executive:
color:    black
drawing style:      spline
doc file: pr_transition

--------------------------------------------------------------------------------

================================================================================
                            transition   idle -> idle
================================================================================
name:     tr_3
condition:          default
executive:
color:    black
drawing style:      spline
doc file: pr_transition

--------------------------------------------------------------------------------

================================================================================
                    transition   idle -> queue_state_change
================================================================================
name:     tr_6
condition:          QUEUE_STATE_CHANGE
executive:
color:    black
drawing style:      spline
doc file: pr_transition

--------------------------------------------------------------------------------
```

```
================================================================================
                Enter Execs for the forced state "pkt_arrival"
================================================================================
pkptr =  op_pk_get (op_intrpt_strm ());


if (upper_mode_flag)
          {
          op_pk_send (pkptr, NONCONT_OUT_STRM);
          }
else
          {
          op_pk_send (pkptr, CONT_OUT_STRM);
          }

--------------------------------------------------------------------------------
================================================================================
                Exit Execs for the forced state "pkt_arrival"
================================================================================
NONE
--------------------------------------------------------------------------------
================================================================================
                        transition   pkt_arrival -> idle
================================================================================
name:     tr_2
condition:
executive:
color:    black
drawing style:      spline
doc file: pr_transition

--------------------------------------------------------------------------------


================================================================================
            Enter Execs for the forced state "queue_state_change"
================================================================================
current_intrpt_src = op_intrpt_stat();

if ((current_intrpt_src == CONT_STAT_STRM) && (op_stat_local_read(current_intrpt_src) == 1.0))
          {
          upper_mode_flag = 1;
          }

if ((current_intrpt_src == NONCONT_STAT_STRM) && (op_stat_local_read(current_intrpt_src) == 0.0))
          {
          upper_mode_flag = 0;
          }

if(fix_data_in_non_cont)
          {
          upper_mode_flag = 1;
          }

--------------------------------------------------------------------------------
================================================================================
             Exit Execs for the forced state "queue_state_change"
================================================================================
NONE
--------------------------------------------------------------------------------
================================================================================
                    transition   queue_state_change -> idle
================================================================================
name:     tr_7
condition:
executive:
color:    black
drawing style:      spline
doc file: pr_transition

--------------------------------------------------------------------------------
```

THIS PAGE INTENTIONALLY LEFT BLANK

INITIAL DISTRIBUTION LIST

1. Defense Technical Information Center
   Ft. Belvoir, Virginia

2. Dudley Knox Library
   Naval Postgraduate School
   Monterey, California

3. Prof. Murali Tummala
   Naval Postgraduate School
   Monterey, California

4. Prof. John McEachen
   Naval Postgraduate School
   Monterey, California

5. Prof. Herschel Loomis
   Naval Postgraduate School
   Monterey, California

6. Prof. Roberto Cristi
   Naval Postgraduate School
   Monterey, California

7. Prof. J. Bret Michael
   Naval Postgraduate School
   Monterey, California

8. CDR Owens Walker
   United States Naval Academy
   Annapolis, Maryland

9. Bob Broadston
   Naval Postgraduate School
   Monterey, California

10. Thomas Kirkpatrick
    SPAWAR SSC ATLANTIC
    Charleston, South Carolina

11. Michael Niermann
    SPAWAR SSC ATLANTIC
    Charleston, South Carolina

12. LCDR Michael Riggins
USSOCOM HQ
MacDill AFB, Florida

13. James Cathcart
National Security Agency
Fort George G. Meade, Maryland

14. Elbert M. Ruiz
National Security Agency
Fort George G. Meade, Maryland

15. John G. Kato
Naval Information Operations Command Suitland
Suitland, Maryland

16. John T. Scott
Naval Information Operations Command Suitland
Suitland, Maryland